commodore Magazine

AÑO I - Núm. 6 - Agosto 1984 - 250 Ptas.

REVISTA INDEPENDIENTE PARA USUARIOS

El misterio del BASIC

Lápices ópticos para todos

Concurso, juegos, aplicaciones

commodore *Magazine*

Sumario

Commodore Magazine es una publicación de Ediciones y Suscripciones, S. A., C/ Bravo Murillo, 377 - Madrid 20. Tel. (91) 733 74 13 / 47 / 63 / 97.

REDACCION
Director:
Alejandro Diges.
Colaboradores:
Aníbal Pardo.
Gumersindo García.
Roberto Menéndez.
Simeón Cruz.
Fernando García.
Manuel Arias.
Diseño:
Ricardo Segura.

EDITORIAL
Presidente:
Fernando Bolín.
Director Editorial:
Norberto Gallego.
Coordinador Editorial:
J. A. Sanz.

ADMINISTRACION
Gerente de Circulación y Ventas:
Luis Carrero.
Suscripciones: Antonio Zurdo.
Producción:
Miguel Onieva.
Publicidad Madrid:
Roberto Rodríguez.
Bravo Murillo, 377.
Madrid-20.
Tel. (91) 733 74 13.
Publicidad Barcelona:
Pelayo, 12.
Tel. (93) 301 47 00, Ext. 27.
Distribuye: SGEL.
Avda. Valdelaparra s/n.
Alcobendas, Madrid.
Imprime: Novograph, S. A., Ctra. de Irún, Km. 12.450 Madrid.
Fotomecánica: Karmat. Pantoja, 10. Madrid.
Depósito Legal: M-6622-1984.

Año 1
Núm. 6

SUSCRIPCIONES

Rogamos dirija toda la correspondencia relacionada con suscripciones o números atrasados a:

commodore *Magazine*

EDISA
López de Hoyos, 141, 5.º
MADRID-2
Tel. 415 97 12

Editorial

Esta edición de la revista queremos que sea un número refrescante para el cálido clima del estío. En un principio pensamos dedicar el espacio de la revista únicamente a programas, pero finalmente decidimos que no vendrían mal algunos artículos. A fin de cuentas el verano también es un buen momento para aprender y repasar temas.

Después de reservar un pequeño espacio, no todo el que quisiéramos, a contestar cartas de lectores, pasamos a un interesante artículo que desvela los misterios de como guarda el ordenador los programas escritos en BASIC. Bueno, en este caso concreto el protagonista elegido es el **C-64.** De ahí el título **El Misterio del BASIC,** no porque sea un misterio, sino más bien porque a gran parte de los usuarios les puede parecer que lo es.

El profesor **Roca Dorda,** de la Escuela Universitaria Po técnica de Cartagena, nos presenta la primera parte de una serie de artículos orientados al funcionamiento y aplicaciones de los lápices ópticos. Acompaña sus textos con esquemas teóricos y prácticos, no en balde su especialidad es la electrónica. Así es factible que los lectores "manitas" sopesen la posibilidad de construirse el suyo.

La serie que no hemos interrumpido es la que nos cuenta como interpretar lo que nos ofrece el manual, esta vez bajo el título de **Visualización de Caracteres,** un tema ameno e interesante para sacarle mayor partido a la máquina.

Incluimos también una encuesta que, cuando nos ofrezca sus interesantes respuestas, nos servirá para acomodar más la revista a los gustos y preferencias generalizados de los lectores.

Por supuesto, las secciones de **Programas** y **Concurso** siguen con su relevancia tradicional en la revista. En esta ocasión sale el gordo, con el resultado del concurso de aplicaciones pensadas para el **Calc Result,** donde la respuesta no ha sido tan masiva como hubiera sido de desear.

No nos queda más que desearos que sigáis disfrutando de las delicias del tórrido verano.

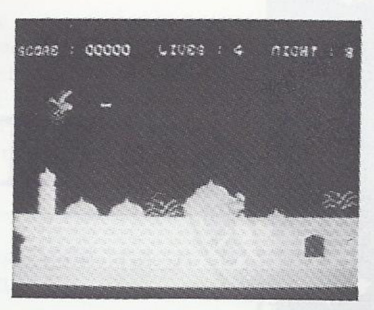
Claves para introducir en el ordenador los programas que aparecen en la revista.

COMO SE VE	COMO SE TECLEA	EFECTO CONSEGUIDO
[illegible]	SHIFT + CLR	(LIMPIA PANTALLA)
[illegible]	CLR	(HOME)
[illegible]	CRSR	(CURSOR ABAJO)
[illegible]	SHIFT + CRSR	(CURSOR ARRIBA)
[illegible]	CRSR	(CURSOR DERECHA)
[illegible]	SHIFT + CRSR	(CURSOR IZQUIERDA)
[illegible]	CTRL + 1	(NEGRO)
[illegible]	CTRL + 2	(BLANCO)
[illegible]	CTRL + 3	(ROJO)
[illegible]	CTRL + 4	(CIAN)
[illegible]	CTRL + 5	(VIOLETA)
[illegible]	CTRL + 6	(VERDE)
[illegible]	CTRL + 7	(AZUL)
[illegible]	CTRL + 8	(AMARILLO)
[illegible]	CTRL + 9	(CARACTER INVERSO)
[illegible]	CTRL + 0	(CARACTER NORMAL)

PROGRAMA: TALES OF THE ARABIAN NIGHTS
TIPO: JUEGO
DISTRIBUIDOR: ABC SOFT
FORMATO: CINTA DE CASSETTE
COMPUTADOR: COMMODORE 64 CON JOYSTICK

Este juego está basado en la leyenda de las mil y una noches. Mientras la princesa Anitra cuenta historias a Saladino, su novio Imrahil debe pasar por mil peligros para resctarla.

El programa sigue la línea del Kong y muchos más representando en diversas pantallas sucesivas pruebas que el príncipe aventurero debe superar. En algunas deberá reunir por orden las letras que forman la palabra "ARABIAN" que se hallan dispersas, en otras ocasiones volará con alfombras o navegará por



ríos cogiendo bolas de fuego que le salen al paso. Estos movimientos se controlan con el *joystick*, que además de las direcciones principales usa el botón de disparo para saltar de un sitio a otro. La dificultad va en aumento progresivo, aunque ya, el primer nivel, es bastante complicado y exige una sincronización perfecta para evitar caer en las garras de los "malos".

El juego posee una fascinación sorprendente, aumentada con música y con un sintetizador de voz que hace que el ordenador hable. Estas dos opciones son eliminables a voluntad para aquellas personas que prefieren los juegos silenciosos.

La cinta viene en una atractiva caja con completas instrucciones y la historia completa tal como la narra la leyenda. Todos estos datos vienen en inglés, pero el importador nos ha asegurado que pronto estarán disponibles en español.

El juego está muy bien hecho y tiene unas características de lo común (sintetizador de voz, carga ultrarrápida) que hacen muy aconsejable su compra.

PUNTUACION:
ADICCION: 7.
PRESENTACION: 9.
GRAFICOS: 6.
ACCION: 7.

PROGRAMA: MAZIACS
TIPO: JUEGO
DISTRIBUIDOR: ABC SOFT
FORMATO: CINTA DE CASSETTE
COMPUTADOR: COMMODORE 64

Básicamente consiste en un labrinto de gran tamaño en el que se ha escondido un tesoro, el jugador parte de la zona central del laberinto y debe encontrar el tesoro y llevarlo al lugar de partida. Por las paredes se encontrará con espadas, comida y prisioneros a los que no puede rescatar pero que le indicarán la dirección en que se halla el tesoro, o si ya lo tiene, la dirección para volver a casa.

Pero no todo va a ser bueno, en el laberinto habitan los MAZIACS, terribles monstruos que le atacan y de los que se puede defender con ayuda de una espada (si no tiene espada el resultado de la pelea es bastante dudoso), cada vez que pelee con uno la espada desaparece y debe buscar otra. En el camino de vuelta, cuando ya tiene el tesoro, las cosas se complican porque no se pueden llevar dos objetos a la vez y cuando nos encontramos con un monstruo tenemos que dejar el tesoro y agarrar una espada para matar al monstruo (si tenemos una espada cerca, claro). En la parte inferior de la pantalla aparecen dos barras indicadoras, la primera señala la fuerza que nos queda, ésta aumenta al comer y disminuye al andar o luchar con un monstruo (disminuyendo mucho más en este segundo caso). La segunda barra indica lo cerca que estamos del tesoro, aunque esta indicación no es fiable al cien por cien ya que podemos estar al lado del tesoro pero separados por una pared que nos obligue a realizar un largo rodeo.

El manejo se puede realizar con el teclado o con el *joystick*

Cuando queramos coger un objeto o interrogar a un prisionero debemos intentar movernos hacia él . Básicamente consiste en mover a nuestro personaje, ya que las luchas con el monstruo, una vez que nos encontramos con él, con totalmente automáticas limitándonos a hacer de espectadores. Además de los movimientos existe otra opción que nos permite ampliar nuestro campo de visión y ver más trozo del laberinto (aunque no se le llega a ver todo entero).

PUNTUACION:
ADICCION: 5.
PRESENTACION: 6.
GRAFICOS: 8.
ACCION: 8.

EL MISTERIO DEL BASIC



Acabamos de conseguir nuestro ansiado ordenador. Inmediatamente leeremos el manual o algún libro sustitutorio, de la mano del cual teclearemos nuestras primeras líneas de programación en BASIC. Una de las cosas más clásicas que se suele teclear es algún corto programa que enseñe al ordenador a llamarnos por nuestro nombre, algo así como:

10 INPUT "COMO TE LLAMAS"; A$

20 PRINT "BIEN"; A$; "VAMOS A JUGAR A..."

Después como es lógico, la complejidad de nuestros desarrollos irá aumentando, en la medida en que prestemos más tiempo y dedicación al ordenador. Poco a poco vamos comprobando como la máquina hace lo que le digamos, siempre que respetemos unas normas para que el programa sea coherente.

Sin embargo muy a menudo surgen dudas en torno a como almacena el ordenador nuestro programa, y sobre todo como lo ejecuta posteriormente. El manual no aclara gran cosa, pues esto no es imprescindible para ser un inmejorable programador. Al igual que tampoco es necesario saber como funciona un reloj para conocer la hora. De todas formas es probable que surja la intriga, y eso es lo que vamos a desvelar a lo largo del artículo.

Al examinar el manual vimos por algún lado el mapa de memoria utilizado en el **64**. También, por algún lado, aparece una leyenda similar a "zona de memoria reservada para el BASIC".

Un mapa de memoria es en sí una representación de como se distribuye la memoria disponible por el ordenador, en función de sus necesidades y prestaciones. Normalmente se representa como un rectángulo en sentido vertical, seccionado por varias líneas horizontales, que a su vez definen distintas áreas. Cada una queda delimitada por dos números (en decimal

```
1000 OPEN3,4
1010 PRINT#3,CHR$(15)"CODIGO        CARACTER"
1020 PRINT#3,CHR$(15)CHR$(10)
1030 FOR X=32 TO 95
1050 Z$=CHR$(X)
1060 PRINT#3,CHR$(15)X;"........";Z$
1070 NEXTX
1080 CLOSE3
```

Figura 1. A. Listado del programa.

Lista de algunos caracteres ASCII.

CODIGO	CARACTER	CODIGO	CARACTER
32		64	@
33	!	65	A
34	"	66	B
35	#	67	C
36	$	68	D
37	%	69	E
38	&	70	F
39	'	71	G
40	(	72	H
41	)	73	I
42	*	74	J
43	+	75	K
44	,	76	L
45	-	77	M
46	.	78	N
47	/	79	O
48	0	80	P
49	1	81	Q
50	2	82	R
51	3	83	S
52	4	84	T
53	5	85	U
54	6	86	V
55	7	87	W
56	8	88	X
57	9	89	Y
58	:	90	Z
59	;	91	[
60	<	92	£
61	=	93	]
62	>	94	↑
63	?	95	←

Figura 1.

o hexadecimal), que corresponden a direcciones reales de la memoria.

Volvamos a ese área reservada a los programas en BASIC. Pues bien, en ellas es donde se guardan todas las líneas de nuestro programa. En el **64** este área comienza normalmente en la dirección 2049 ($801) y se extiende hasta la 40959 ($9FFF).

En realidad el inicio está en la dirección 2048, que contendrá el número cero a modo de indicador.

TOKEN	PALABRA CLAVE	TOKEN	PALABRA CLAVE	TOKEN	PALABRA CLAVE
128	......END	153	......PRINT	178	......=
129	......FOR	154	......CONT	179	......<
130	......NEXT	155	......LIST	180	......SGN
131	......DATA	156	......CLR	181	......INT
132	......INPUT#	157	......CMD	182	......ABS
133	......INPUT	158	......SYS	183	......USR
134	......DIM	159	......OPEN	184	......FRE
135	......READ	160	......CLOSE	185	......POS
136	......LET	161	......GET	186	......SQR
137	......GOTO	162	......NEW	187	......RND
138	......RUN	163	......TAB(	188	......LOG
139	......IF	164	......TO	189	......EXP
140	......RESTORE	165	......FN	190	......COS
141	......GOSUB	166	......SPC(	191	......SIN
142	......RETURN	167	......THEN	192	......TAN
143	......REM	168	......NOT	193	......ATN
144	......STOP	169	......STEP	194	......PEEK
145	......ON	170	......+	195	......LEN
146	......WAIT	171	......-	196	......STR$
147	......LOAD	172	......*	197	......VAL
148	......SAVE	173	....../	198	......ASC
149	......VERIFY	174	......↑	199	......CHR$
150	......DEF	175	......AND	200	......LEFT$
151	......POKE	176	......OR	201	......RIGHT$
152	......PRINT#	177	......>	202	......MID$

Figura 2. Lista de token y palabras reservadas.

```
1000 OPEN3,4
1003 PRINT#3,CHR$(15)"TOKEN      PALABRA"
1005 PRINT#3,CHR$(15)"           CLAVE"
1007 PRINT#3,CHR$(15)CHR$(10)
1010 A=41117
1020 X=127
1030 X=X+1
1040 PRINT#3,CHR$(15)X;"......";
1050 A=A+1
1060 B=PEEK(A)
1070 IFB>128THENGOTO2000
1080 C$=CHR$(B)
1090 PRINT#3,CHR$(15)C$;
1100 IFA=203 THEN CLOSE3
1110 GOTO1050
2000 B=B-128
2010 C$=CHR$(B)
2020 PRINT#3,CHR$(15)C$
2030 GOTO1030
```

Figura 2. A. Listado del programa.

Entre las direcciones 40960 (AOOO) y 49151 ($BFFF) se halla el BASIC en ROM, sobre el que hablaremos enseguida. Es lo que hace que este lenguaje sea algo utilizable en el ordenador. Una última zona ocupada tambien por memoria *ROM* contiene información muy útil para que el **64** actúe correctamente, es el *KERNAL*, sistema operativo consistente en una serie de programas, que dan coherencia al funcionamiento de la máquina. En realidad el *KERNAL* es una tabla que recoge los saltos que deben ser realizados hacia las diversas rutinas del sistema operativo. Su ubicación abarca desde la dirección 57344 ($EOOO) hasta la 65535 ($FFFF).

El BASIC en *ROM* contiene muchas rutinas separadas, pero podemos agruparlas entre sí en dos sistemas principales: el **editor** y el **intérprete.**

```
10 REM***COMMODORE 64
20 PRINT"PRUEBA"
```

Figura 3. Líneas de ejemplo.

El **editor** trabaja de forma visible en la pantalla, manejando todo el trabajo ingrato de modificar, insertar y borrar cosas en la línea de programa que más nos guste. En otras palabras, el **editor** controla la entrada de programas, sean líneas sin número (comandos directos) o con él. Siempre que el ordenador no esté ejecutando algún programa, es el **editor** quien tiene el control de la máquina.

El **intérprete** entra en acción en cuanto teclamos el comando *RUN*, conservando el control hasta el final de la ejecución del programa, o hasta que decidamos detenerlo por alguno de los procedimientos de paro de la misma.

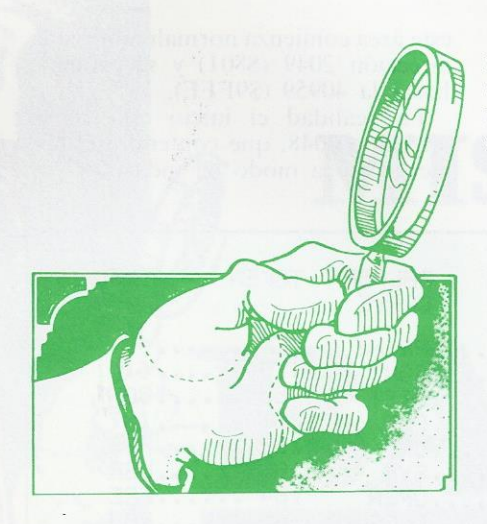
Todas y cada una de las líneas de programa que escribamos se guardan en la zona de memoria *RAM* asignada al BASIC siguiendo el mismo formato, independientemente de su dirección de comienzo y su longitud.

Como todo el mundo sabe, el BASIC se caracteriza, entre otras cosas, por disponer de un nutrido

conjunto de **palabras clave,** de las cuales quizas la más utilizada sea *PRINT*.

El **64,** como todos los ordenadores, almacena la información en base a unos y ceros, donde cada combinación tiene una correspondencia. Una agrupación de 8 es el conocido byte. Un byte sirve para representar muchas cosas, también caracteres alfanuméricos, símbolos aritméticos y gráficos, números, etc. Si cada letra puede ser representada mediante un byte, *PRINT* necesitaría de 5 bytes. Realmente es una forma inútil de malgastar memoria. Si pudiéramos hacer corresponder a cada palabra clave un byte diferente, el ahorro en un programa largo sería notable, sobre todo si utilizamos muy a menudo *RESTORE* o *RETURN*. Por otro lado, el **intérprete** BASIC necesita ir leyendo una por una todas las líneas de programa para ir ejecutándolo. Eso implica que lea letra por letra (byte por byte). Nuevamente, si consiguiéramos representar las palabras clave con un solo byte, la velocidad de ejecución de los programas sería mucho más rápida. También habría que contar con que el **intérprete** debe comparar cada palabra que aparece en la línea con una lista de todas las **palabras clave,** a la que se llama **lista de palabras reservadas.** Dejamos de jugar a las suposiciones, esto es lo que hace en la realidad el ordenador, traduce cada **palabra clave** a un solo byte equivalente. ¿Pero cuándo lo hace? Hagamos una primera consideración. Mientras el control del ordenador está en manos del **editor,** dispone de mucho tiempo libre, pues su principal misión consiste en comunicarse con el hombre por medio del teclado y la pantalla, para lo cual no necesita hacer grandes esfuerzos. Entonces, ¿por qué no dejarle que haga la tarea de ir convirtiendo las palabras clave en un solo byte a medida que las tecleamos? Al fin y al cabo el **intérprete** está normalmente mucho más sobrecargado intentando que el programa corra lo mas aprisa que sea posible.

La **palabra clave,** una vez codificada, recibe el nombre de *token.* Así por ejemplo un comando *tokenizado* es el byte equivalente a ese comando.



Los *token* son generados por una rutina específica, que es llamada por el **editor** cada vez que presionamos la tecla *Return,* después de haber tecleado una línea de programa, sea numerada o de ejecución inmediata (sin numerar). Siempre que introducimos una línea en el ordenador, ésta va almacenándose en una pequeña área de memoria reservada para este fin, llamada *buffer* de entrada. La tecla *Return* produce que las palabras contenidas en este *buffer* sean comparadas con las **palabras clave** de la **lista de palabras reservadas,** sustituyéndose por su equivalente *tokenizado,* antes de ser depositada la línea en la memoria reservada a los pro-

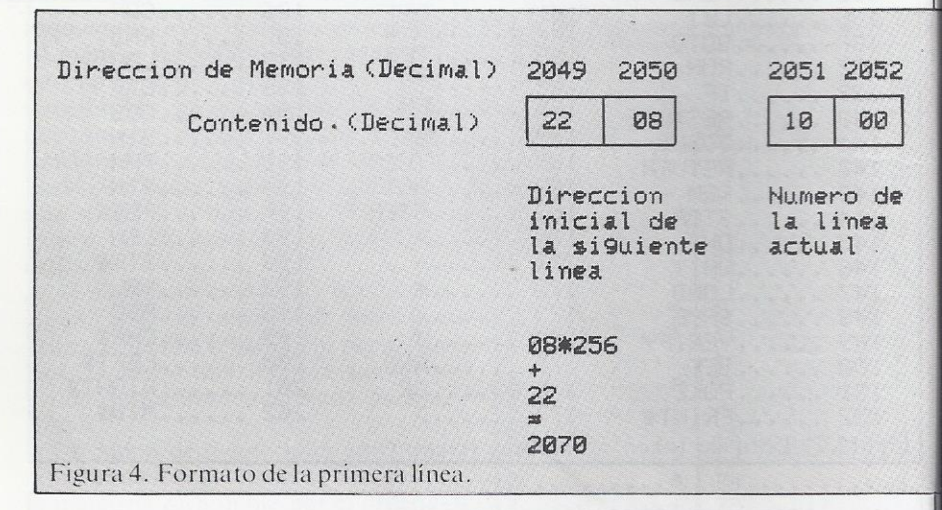


Figura 4. Formato de la primera línea.

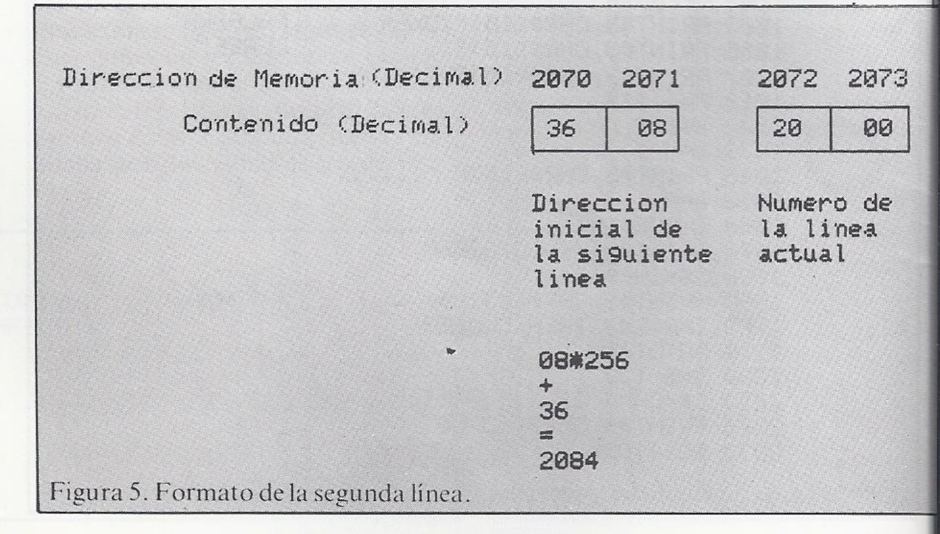


Figura 5. Formato de la segunda línea.

gramas. Como la comparación debe hacerse de un modo ordenado con respecto a la lista cuando se encuentra una palabra, se utiliza como valor del *token* el número correspondiente a la posición que ocupa esa **palabra reservada** en la **lista de palabras reservadas.** Pero una vez *tokenizada*, ¿cómo distingue el ordenador, si está tratando con una palabra clave, un número, un carácter, etc.? Pues muy sencillo, en el momento de la *tokenización* se pone uno al bit de mayor peso del byte (es el equivalente a sumarle 128 en decimal).

El resto de los caracteres contenidos en la línea escrita (todavía en el *buffer* de entrada) son almacenados definitivamente con su equivalente en código ASCII.

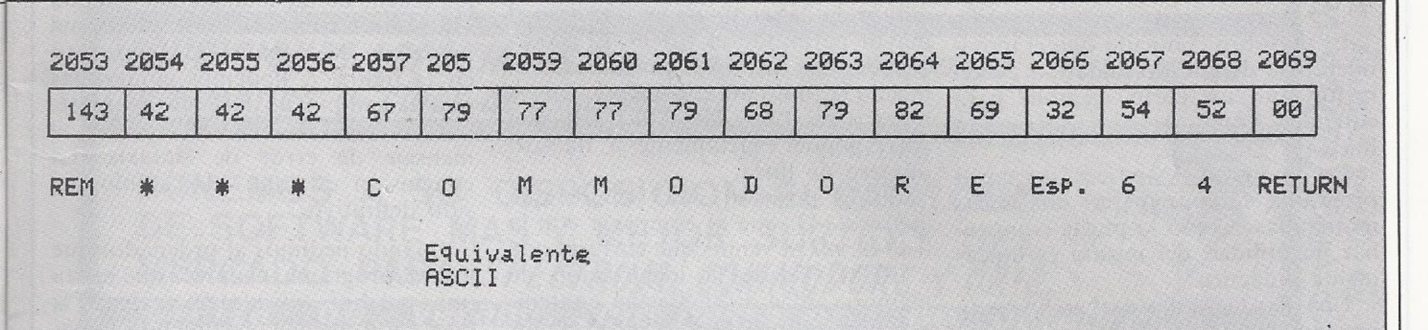
En la figura 1 aparecen los caracteres utilizados por el ordenador en la escritura de programas (se han omitido los gráficos y especiales) y el programa utilizado para su obtención (figura 1 A).

La figura 2 muestra la lista de palabras BASIC reservadas y su *token* equivalente, en decimal, a la izquierda (como es obvio, a cada número decimal menor de 255 se le puede hacer corresponder un solo byte).

En 2.A. tenemos el listado del programa utilizado para obtener la lista de *token*. La línea 1070 compara los bytes, para encontrar uno mayor de 128, que indica que estamos ante una nueva **palabra clave.** La lista de palabras reservadas comienza en la dirección memoria 41117.

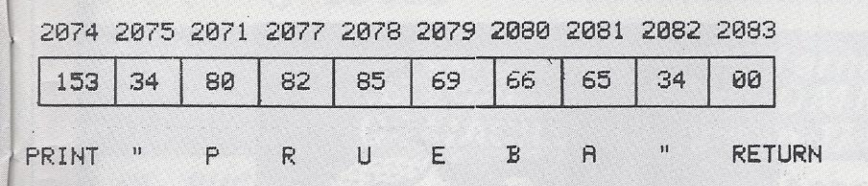
Pronto se hará evidente que en las figuras 1 y 2 aparecen símbolos comunes, concretamente +, -, *, /, ↑, >, = y <.

En la lista de *token* se trata de operadores aritméticos y lógicos. Sin embargo, en la de caracteres son tratados simplemente como eso. Por tanto en ambos listados tienen diferente código, aunque su aspecto es el mismo al ser impar.



2053	2054	2055	2056	2057	205	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069
143	42	42	42	67	79	77	77	79	68	79	82	69	32	54	52	00
REM	*	*	*	C	O	M	M	O	D	O	R	E	Esp.	6	4	RETURN

Equivalente
ASCII



2074	2075	2071	2077	2078	2079	2080	2081	2082	2083
153	34	80	82	85	69	66	65	34	00
PRINT	"	P	R	U	E	B	A	"	RETURN

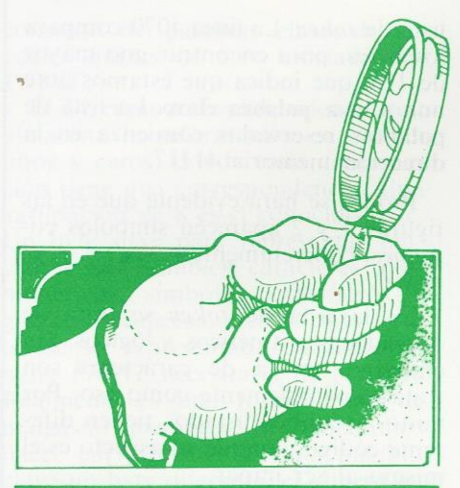
Equivalente
ASCII

```
10 POKE 2053,153
20 POKE 2074,143
```

Figura 6. Cambio de los *token*.

```
10 PRINT***COMMODORE 64
20 REM"PRUEBA"
```

Figura 7. Líneas con los *token* intercambiados.



Es de sobra conocido que el movimiento se demuestra andando. Nada mejor que un corto ejemplo para entender cómo se almacena cada línea.

En la figura 3 aparece un corto programa compuesto por dos líneas numeradas. Como se puede comprobar, la utilidad del mismo es únicamente didáctica.

Una vez tecleadas ambas líneas (y presionada la tecla Return), el ordenador las almacena en la memoria, comenzando por la dirección 2049.

La figura 4 muestra cómo ha sido almacenada la primera línea y la información contenida en cada dirección sucesiva. Ahora es cuando podemos comenzar a hablar de formatos. Los dos primeros bytes (direcciones 2049 y 2050) contienen una importante información: el lugar donde se comienza a guardar la siguiente línea de programa. La necesidad de utilizar dos bytes se hace patente desde el momento en que (recordemos) con un solo byte sólo podemos obtener 256 posibilidades distintas. Sin embargo, el microprocesador puede acceder hasta a 64 K direcciones, que es el número de posibilidades que proporcionan dos bytes (2^{16}—1).

Para convertir un número hexadecimal en su equivalente decimal, se recurre a un mecanismo de cálculo idéntico al empleado en la notación decimal, donde hay dígitos en la posición de las unidades, de las decenas, las centenas, etc. En el presente caso podríamos hacer una comparación útil con un número de dos dígitos (decenas y centenas), donde el primer byte sería equivalente a las unidades y el segundo a las decenas, sólo que en un byte hay 256 posibilidades y sólo diez en una decena. En otras palabras, el número 99 se compone de 9 + 9 × 10, en el ejemplo, 22-8 es igual a 22 + 256 × 8 = 2 070. Observemos que esta línea termina en la 2 069.

Los dos siguientes bytes guardan la primera información que afecta a la línea de programa en sí, es el número de línea que hemos asignado. Nuevamente se requieren dos bytes para que el número de línea pueda ser mayor de 255, aunque en este caso el mayor permitido es 63999. El procedimiento de cálculo es exactamente el mismo seguido anteriormente. En este caso es: 10 + 256 × 0 = 10, que corresponde exactamente a nuestro número de línea.

En la posición 2053 vemos que aparece 143, que al comparar con la lista de *token* vemos que corresponde a REM. Desde la localización de memoria 2054 a 2068 están contenidos los equivalentes ASCII de los caracteres del resto de la línea. La posición 2069 actúa como separador con la siguiente línea, pero también es el código correspondiente a Return. Conviene no confundirlo con el espacio en blanco, que es el 32 en el código ASCII. Tampoco con el cero, cuyo código es 48.

En la figura 5 tenemos la línea 20. Aunque se trata de un tipo diferente de sentencia, el formato utilizado para guardarla es el mismo: comienzo de la siguiente línea (2084), número de línea (20), *token* correspondiente a PRINT (153), caracteres ASCII y Return (00).

Este ejemplo toma dos de los *token* más empleados. De todas maneras, si una línea dispone de varias sentencias, el carácter dos puntos será quien indique los límites y finalmente aparecerá el clásico 0 de Return, y dos ceros más de final de programa (el comienzo de la siguiente línea que no existe). Hagamos un experimento. Introduciendo los POKE indicados en la figura 6, cambiaremos entre sí los *token*, de tal manera que cuando volvamos a escribir LIST el programa aparecerá en la pantalla como se ve en la figura 7. Podemos comprobar que el cambio no ha sido aparente, pues si tecleamos RUN, aparece el mensaje de error de sintaxis. La conclusión es que el cambio ha sido definitivo.

Cuando pedimos al ordenador que liste el programa, cada vez que exista una palabra *tokenizada* se hace la operación inversa, escribiendo la palabra como en un principio fue tecleada.

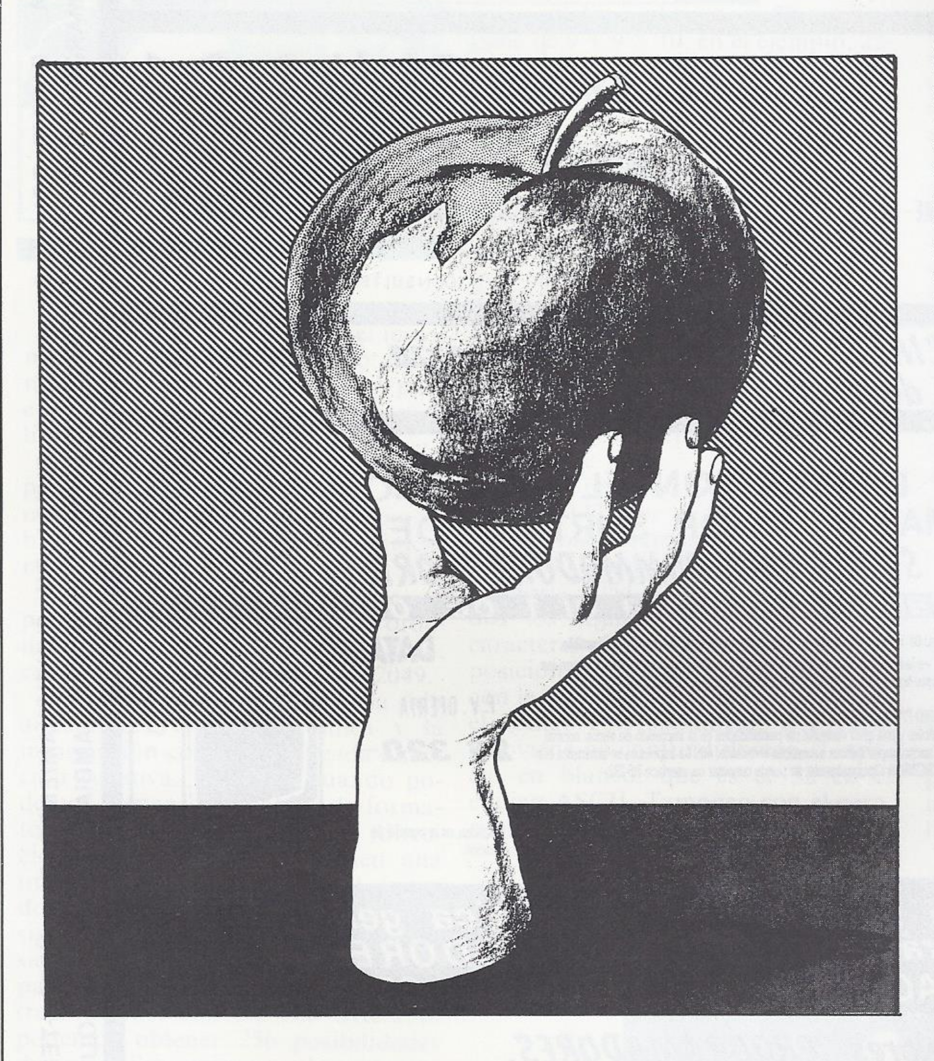
En el próximo número veremos algunos detalles particulares sobre como son ejecutados los programas, particularmente el funcionamiento de una útil rutina en código máquina, llamada CHRGET.



Programas

Manzamón

VIC 20



Manzamón es el monstruo de las manzanas que avanza para zamparse la manzana, destruyendo los bloques defensivos que se oponen a su avance. El jugador tiene el mando de un láser, en la línea inferior de la pantalla, que es la única arma capaz de acabar con el monstruo. El láser se controla mediante tres teclas del teclado que se detallan al comienzo del programa, en las instrucciones. Al finalizar, el programa da los puntos conseguidos.

```
10 REM *************
20 REM *           *
30 REM * MANZAMON *
40 REM *           *
50 REM *************
100 PRINT"J":P=0:POKE36879,127:Z=150:GOSUB360
110 PRINT"J":POKE56,28:POKE52,28:POKE650,128:L=8175:S=0:POKE36878,15:
:O=0\ POKE36877,0
120 POKE36869,255:Z=Z-10:IFZ<30THENZ=30
```

```
130 FORT=21TO470
140 G=INT(6*RND(1)):POKE36874,150
150 POKE7680+T+30720,0:POKE7680+T,209:POKE7680+T,32
160 IFG=2THENPOKE7680+T,102+128:POKE7680+T+30720,0:POKE36874,0
170 NEXTT:POKE36874,0:POKE8185,62:POKE8185+30720,4
180 A=7680:F=1
190 A=A+F:POKE36876,230:IFA=8185THENPOKEA,63:POKEA+30720,0:POKEE,32:GOTO540
200 IFPEEK(Z)=102+128ANDF=1THENA=A+21:F=-1:POKE36875,235
210 IFPEEK(A)=102+128ANDF=-1THENA=A+23:F=1:POKE36875,235
220 IFPEEK(A)=60THEN540
230 POKEA,63:POKEA+30720,2
240 IFS=1THEN490
250 FORT=0TOZ:NEXTT:POKE36875,0:POKE36876,0
260 POKEA,32:GETA$
270 IFA$="Z"THENPOKEL,32:L=L-1
280 IFA$="/"THEN POKEL,32:L=L+1
290 IFL=8163THENL=8164
300 IFL=8185THENL=8184
310 IFA$=" "ANDS=0THENS=1:E=L
320 IFS=0THENPOKE36877,0
330 IFO=1THENPOKE36877,0:O=0
340 POKEL,60:POKEL+30720,0
350 GOTO190
360 PRINT"██████████MANZAMON"
370 PRINT"███████---------"
380 PRINT
390 PRINT"█'?'=DERECHA"
400 PRINT"██'Z'=IZQUIERDA"
410 PRINT"████'ESPACIO'=FUEGO"
420 PRINT"█████████ESPERA..."
430 FORT=0TO511:POKE7168+T,PEEK(32768+T):NEXTT
440 FORT=0TO3
450 FORI=0TO7:READB:POKE7648+T*8+I,B:NEXTI:NEXTT:POKE36869,255
460 RETURN
470 DATA16,16,16,16,16,126,126,255,0,16,0,16,0,16,0,16
480 DATA7,44,24,126,255,255,126,44,126,219,255,255,169,129,149,126
490 E=E-22:IFE<7680THENS=0:POKEE+22,32:GOTO250
500 POKE36877,128+125
510 IFPEEK(E)=102+128THENP=P-10:S=0:POKEE,32:POKEE+22,32:POKE36877,150:O=1:
    GOTO2 50
520 IFPEEK(E)=63THENP=P+500:POKE36877,0:POKE36876,0:GOTO110
530 POKEE,61:POKEE+30720,2:POKEE+22,32:POKEL,60:POKEL+30720,0:GOTO250
540 POKE36876,0:POKE36875,0:POKE36877,0:POKE36869,240
550 PRINT"█"
560 PRINT"███████PUNTOS:";P
570 IFP>HPTHENHP=P
580 PRINT"██RECORD:";HP
590 PRINT"██OTRA VEZ (S/N)? "
600 GETA$:IFA$="S"THENZ=150:GOTO110
610 IFA$<>"N"THEN600
620 PRINT"█":PRINT"█████ADIOS.":FORT=1TO1500:NEXTT
630 PRINT"██":POKE36879,27:END
```

Programas

VIC-20 + EXPANSION DE 16K

Comecocos

Aquí está el que, sin duda alguna, ha sido el rey entre los programas de juegos durante mucho tiempo: el **COMECOCOS**, que os presentamos en una sencilla versión adaptada al **VIC-20** y que esperamos que os haga pasar buenos ratos. El programa está escrito en *BASIC* por lo que no es demasiado rápido, pero ello no lo hace menos entretenido ni mucho menos. El Comecocos se maneja con cuatro teclas para moverlo en cualquier dirección por el laberinto. El juego consiste en comerse todos los puntos que hay por los pasillos del laberinto pero teniendo cuidado para no ser comido por los monstruos que andan por allí. Inicialmente se dispone de cinco vidas, que se van perdiendo cada vez que nos topamos con un monstruo. El programa termina cuando se acaban las vidas, o bien cuando el comecocos se ha zampado todos los puntos. Las instrucciones de juego van incluidas al comienzo del programa.

Variables utilizadas:

A continuación damos una lista de las variables utilizadas en el programa y lo que representa cada una de ellas:

VR Vidas restantes.
MO Carácter Monstruo.
PM Carácter Pacman (Comecocos).
PP Posición Pacman (Comecocos).
SC Puntuación.
MP% Posición Monstruo.
PA Pared del laberinto.

```
1 REM *********************
2 REM *    COMECOCOS      *
3 REM *                   *
4 REM *COMMODORE MAGAZINE*
5 REM *********************
10 DIMMP%(4):
20 VR=5
25 POKE 36879,141
30 PRINT"▒▒▒      *************"
35 PRINT"        COMECOCOS"
40 PRINT"      *************
45 PRINT"▒▒▒     COMETE LOS PUNTOS"
50 PRINT"▒▒▒▒*=COMECOCOS":PRINT"▒▒▒▒X=MONSTRUO"
55 PRINT"▒▒▒▒PARA MOVERSE"
56 PRINT"▒▒▒▒============="
60 PRINT"▒M=DERECHA  F=ARRIBA"
65 PRINT"▒B=IZQUIERDA S=ABAJO"
70 PRINT"▒▒▒PULSA UNA TECLA▒"
100 GET F$:IFF$="" THEN 100
195 POKE36879,91:PRINT"▒";
196 IF VR=0 THEN GOTO5000
200 RESTORE:FORI=4096 TO4601
205 FOR J=1TO23
210 IFP=0THEN READA,P
220 POKEI,A
230 P=P-1
240 NEXTI
250 MO=24:PA=160:PM=42:PP=4525:SC=0
253 MP%(1)=4166:MP%(2)=4180:MP%(3)=4386:MP%(4)=4400
255 GOSUB500
260 FOR L=1 TO4:POKEMP%(L),MO:NEXT
261 IF SC=129THEN GOTO5500
262 FORDD=1TO4
263 GETA$:IF A$=""THEN270
265 GOSUB540
270 Y=INT(RND(1)*4+1)
280 IFY=1THENX=1:GOTO320
```

```
290 IFY=2THENX=-22:GOTO320
300 IFY=3THENX=-1:GOTO320
310 IFY=4THENX=22:GOTO320
320 IFPEEK(MP%(DD)+X)=PATHEN 270
322 IFPEEK(MP%(DD)+X)=MOTHEN270
325 IFPEEK(MP%(DD)+X)=PMTHEN400
327 IF PEEK(MP%(DD)+X)=46THEN POKEMP%(DD),46
330 IF PEEK(MP%(DD)+X)=32THENPOKEMP%(DD),32
335 PRINT"▓VIDAS";VR;" PUNTOS";SC
340 MP%(DD)=MP%(DD)+X
344 POKE36878,15:FORQQ=1TO5:POKE36877,150:NEXT
345 POKE36878,0:POKE36877,0
347 NEXTDD
350 GOTO260
400 PRINT"♥":POKE36879,141:FORI=1TO6:PRINT" "TAB(I)"CHOMP":NEXT:VR=VR-1
410 FORT=1TO1000:NEXT
420 GOTO195
500 POKEPP,PM
510 RETURN
520 POKEPP,32
530 RETURN
540 IFA$="F"THENZ=-22:GOTO590
550 IFA$="S"THENZ=22:GOTO590
560 IFA$="M"THENZ=1:GOTO590
570 IFA$="B"THENZ=-1:GOTO590
580 RETURN
590 IFPEEK(PP+Z)=PATHEN RETURN
593 IFPEEK(PP+Z)=MOTHEN GOTO400
595 IFPEEK(PP+Z)=46THENSC=SC+1
596 FOR AA=1TO5:POKE36878,15:POKE36876,250:NEXT:POKE36878,0:POKE36876,0
600 GOSUB 520
610 PP=PP+Z
620 GOSUB 500
630 RETURN
1100 DATA32,22,32,22,32,3,160,17
1110 DATA32,2,32,3,160,1,32,1
1120 DATA46,13,32,1,160,1,32,2
1130 DATA32,3,160,1,46,1,160,1
1140 DATA32,1,160,4,46,1,160,4
1150 DATA32,1,160,1,46,1,160,1
1160 DATA32,2,32,3
1170 DATA160,1,46,1,160,1,46,11
1180 DATA160,1,46,1,160,1,32,2
1190 DATA32,3,160,1,46,1,160,6
1200 DATA46,1,160,6,46,1,160,1
1210 DATA32,2,32,3,160,1,46,4
1220 DATA32,1,46,1,160,1,46,1
1230 DATA160,1,46,1,32,1,46,4
1240 DATA160,1,32,2,32,3,160,1
1250 DATA46,1,160,4,46,1,160,1
1260 DATA46,1,160,1,46,1,160,4
1270 DATA46,1,160,1,32,2,32,3
1280 DATA160,1,46,1,160,4,46,1
1290 DATA160,1,46,1,160,1,46,1
1300 DATA160,4,46,1,160,1,32,2,32,3
1310 DATA160,1,46,1,160,4,46,1
1320 DATA160,1,46,1,160,1,46,1
```

Programas

Viene de la página anterior

```
1330 DATA160,4,46,1,160,1,32,2
1340 DATA32,3,160,1,46,4,32,1
1350 DATA46,1,160,1,46,1,160,1
1360 DATA46,1,32,1,46,4,160,1
1370 DATA32,2,32,3,160,1,46,1
1380 DATA160,1,46,4,160,1,46,1
1390 DATA160,1,46,4,160,1,46,1
1400 DATA160,1,32,2,32,3,160,1
1410 DATA46,3,160,2,46,5,160,2
1420 DATA46,3,160,1,32,2,32,3
1430 DATA160,3,46,1,160,9,46,1
1440 DATA160,3,32,2,32,3,160,3
1450 DATA46,4,160,3,46,4,160,3
1460 DATA32,2,32,3,160,6,46,1
1470 DATA160,3,46,1,160,6,32,2
1480 DATA32,3,160,1,46,6,160,3
1490 DATA46,6,160,1,32,2,32,3
1500 DATA160,1,46,1,160,13,46,1
1510 DATA160,1,32,2,32,3
1520 DATA160,1,46,7,32,1,46,7
1530 DATA160,1,32,2,32,3,160,17
1540 DATA32,3,32,22,32,21
1550 DATA32,0
5000 POKE36879,141:PRINT"SE ACABARON LAS VIDAS"
5010 PRINT"⊠HAS MUERTO"
5020 PRINT"⊠LO SIENTO MUCHO"
5025 POKE36878,15
5030 FORPP=255TO128STEP-1
5040 POKE36876,PP
5050 NEXT
5060 POKE36879,27:POKE36876,0:POKE36878,0
5070 END
5500 PRINT"ℑ":POKE36878,15
5510 POKE36879,141:PRINT"BRAVO..."
5520 PRINT"⊠LO HAS CONSEGUIDO"
5530 PRINT"⊠Y TE QUEDAN"
5540 PRINT"⊠";VR;"VIDAS"
5550 FORTT=128TO255
5560 POKE36875,TT
5570 NEXT
5580 POKE36879,27:POKE36878,0:POKE36875,0
5590 END
```



CBM 64

Renumerador

Algunos lectores nos han escrito preguntándonos si no tendríamos por ahí, a mano, una buena rutina para renumerar programas. Pues bien, sí la tenemos y aquí os la presentamos, tanto a los que nos la habéis pedido como al resto de nuestros lectores, incluidos aquellos que no saben lo que es una rutina para renumerar programas.

Para estos últimos diremos que RENUMERADOR es un programa que permite modificar todos los números de línea de las líneas de un programa haciendo además que el salto entre dos líneas consecutivas sea constante en todo el programa. Por ejemplo y para aclarar un poco las cosas supongamos que tenemos un programa cuyos números de línea son **1, 3, 7, 8, 9, 23, 25...** pues bien, con **RENUMERADOR** podemos cambiarlos automáticamente por **10, 20, 30, 40, 50, 60...** o, en lugar de saltar de **diez** en **diez** podemos saltar de **cinco** en **cinco** o de **cien** en **cien**.

Además, esto es lo más importante de una rutina para renumerar, ella sola se encarga de modificar todos los valores de los *GO TO* y de los *GOSUB* para que el programa renumerado funcione igual de bien que el programa original.

El programa de más abajo se encarga de cargar la rutina, que está escrita en lenguaje máquina, en la memoria del ordenador. Para ello hay que copiar el programa con cuidado, para no equivocarse con las *DATA* y una vez comprobado, y después de guardarlo en el cassete por si hay problemas, hay que escribir *RUN*. El programa incluye dos controles para saber si nos hemos equivocado al copiarlo. Si aparece la fracesita SUMA ERRONEA significa que algún número de alguna DATA está mal, mientras que si aparece NUMERO DE VALORES ERRONEO quiere decir que nos hemos comido algún número. En cualquier caso , cuando el programa esté correcto, aparecerá al poco tiempo la frase *UTILIZA SYS* (50505), **INICIO, INCREMENTO.** ESto quiere decir que la rutina está bien cargada en memoria. Ahora podemos escribir *NEW*, ya que, aunque borremos el programa, la rutina en lenguaje máquina sigue en memoria. En este momento podemos escribir cualquier programa, o cargarlo desde cinta o desde el diskette. Para renumerarlo sólo hay que escribir *SYS* (50505), inicio, incremento, donde inicio es el primer número de línea mientras que incremento es el salto entre líneas. Por ejemplo escribiendo *SYS* (50505), 10, 10 al cabo de un momento, el programa habrá cambiado sus números de línea siendo el primero 10 y saltando de diez en diez.

Esperamos que esta rutina os sea de la mayor utilidad a la hora de dejar bonitos los programas, o cuando necesitéis meter líneas entre medias y no tengáis sitio para hacerlo.

Ah, se nos olvidaba, si en en el programa original hay algún *GO TO* o *GOSUB* a una línea que no exista, la rutina de renumerar pone como número de línea **65535**, lo que permite detectar fácilmente errores de este tipo.

```
10 REM ***********************
15 REM *    RENUMERADOR      *
20 REM * COMMODORE MAGAZINE *
25 REM ***********************
30 REM
35 I=50505:T=0:E=0
40 READA:IFA=-1THEN55
45 POKEI,A:I=I+1
50 T=T+A:GOTO40
55 IFT<>52549THEN PRINT"██SUMA ERRONEA":E=1
60 IFI<>50928THENPRINT"██NUMERO DE VALORES ERRONEO":E=1
65 IF E=1 THEN GOTO80
70 PRINT"███UTILIZA SYS(50505),INICIO,INCREMENTO"
75 PRINT"█PARA RENUMERAR CUALQUIER PROGRAMA"
80 END
85 DATA32,253,174,32,107,169,165
90 DATA20,133,53,165,21,133,54
95 DATA32,253,174,32,107,169,165
100 DATA20,133,49,165,21,133,50
105 DATA32,142,166,32,201,198,32
110 DATA201,198,208,33,32,2,198
115 DATA32,201,198,32,201,198,208
120 DATA3,76,212,198,32,201,198
125 DATA165,99,145,122,32,201,198
```

Programas

Viene de la página anterior

```
130 DATA165,98,145,122,32,13,198
135 DATA240,226,32,201,198,32,201
140 DATA198,32,201,198,201,34,208
145 DATA11,32,201,198,240,197,201
150 DATA34,208,247,240,238,170,240
155 DATA188,16,233,162,4,221,235
160 DATA198,240,5,202,208,248,240
165 DATA221,165,122,133,59,165,123
170 DATA133,60,32,115,0,176,211
175 DATA32,107,169,32,32,198,165
180 DATA60,133,123,165,59,133,122
185 DATA160,0,162,0,189,0,1
190 DATA240,17,72,32,115,0,144
195 DATA3,32,82,198,104,160,0
200 DATA145,122,232,208,234,32,115
205 DATA0,176,8,32,97,198,32
210 DATA121,0,144,248,201,44,240
215 DATA186,208,152,165,53,133,99
220 DATA165,54,133,98,76,142,166
225 DATA165,99,24,101,49,133,99
230 DATA165,98,101,50,133,98,32
235 DATA201,198,208,251,96,32,2
240 DATA198,32,201,198,32,201,198
245 DATA208,8,169,255,133,99,133
250 DATA98,48,14,32,201,198,197
255 DATA20,208,16,32,201,198,197
260 DATA21,208,12,162,144,56,32
265 DATA73,188,76,223,189,32,201
270 DATA198,32,13,198,240,209,32
275 DATA114,198,230,251,32,165,198
280 DATA230,45,208,2,230,46,96
285 DATA32,114,198,198,251,32,141
290 DATA198,165,45,208,2,198,46
295 DATA198,45,96,32,124,198,160
300 DATA0,132,17,132,251,96,165
305 DATA122,133,34,165,123,133,35
310 DATA165,45,133,36,165,46,133
315 DATA37,96,164,17,200,177,34
320 DATA164,251,200,145,34,32,190
325 DATA198,208,1,96,230,34,208
330 DATA236,230,35,208,232,164,17
335 DATA177,36,164,251,145,36,32
340 DATA190,198,208,1,96,165,36
345 DATA208,2,198,37,198,36,76
350 DATA165,198,165,34,197,36,208
355 DATA4,165,35,197,37,96,160
360 DATA0,230,122,208,2,230,123
365 DATA177,122,96,32,51,165,165
370 DATA34,166,35,24,105,2,133
375 DATA45,144,1,232,134,46,32
380 DATA89,166,76,116,164,0,137
385 DATA138,141,167,-1
```



SU PROGRAMA PARA CUALQUIER SISTEMA COMMODORE PUEDE HACERLE GANAR 5.000 PTAS.

EL PRESENTE CONCURSO ESTA ABIERTO A TODOS NUESTROS LECTORES Y SU PARTICIPACION E INSCRIPCION ES GRATUITA. LEA LAS BASES DEL CONCURSO

- NO SE ESTABLECEN LIMITACIONES EN CUANTO A EXTENSION, TEMA ELEGIDO O MODELO DE ORDENADOR
- LOS CONCURSANTES DEBERAN ENVIARNOS A LA DIRECCION QUE FIGURA AL PIE, EL CASSETTE O DISKETTE CONTENIENDO EL PROGRAMA, UNA EXPLICACION DEL MISMO Y, AL SER POSIBLE, UN LISTADO EN PAPEL DE IMPRESORA. SE PODRAN ENVIAR TANTOS PROGRAMAS COMO SE DESEE
- LOS PROGRAMAS, PREVIA SELECCION, SERAN PUBLICADOS EN LA REVISTA, OBTENIENDO TODOS ELLOS 5.000 PTAS
- LA DECISION SOBRE LA PUBLICACION O NO DE UN PROGRAMA CORRESPONDE UNICAMENTE AL JURADO NOMBRADO AL EFECTO POR "COMMODORE MAGAZINE", SIENDO SU FALLO INAPELABLE
- LOS CRITERIOS DE SELECCION SE BASARAN EN LA CREATIVIDAD DEL TEMA ELEGIDO Y LA ORIGINALIDAD Y/O SENCILLEZ EN EL METODO DE PROGRAMACION GLOBAL
- ENVIAR A: CONCURSO COMMODORE MAGAZINE

ORIGINALIDAD

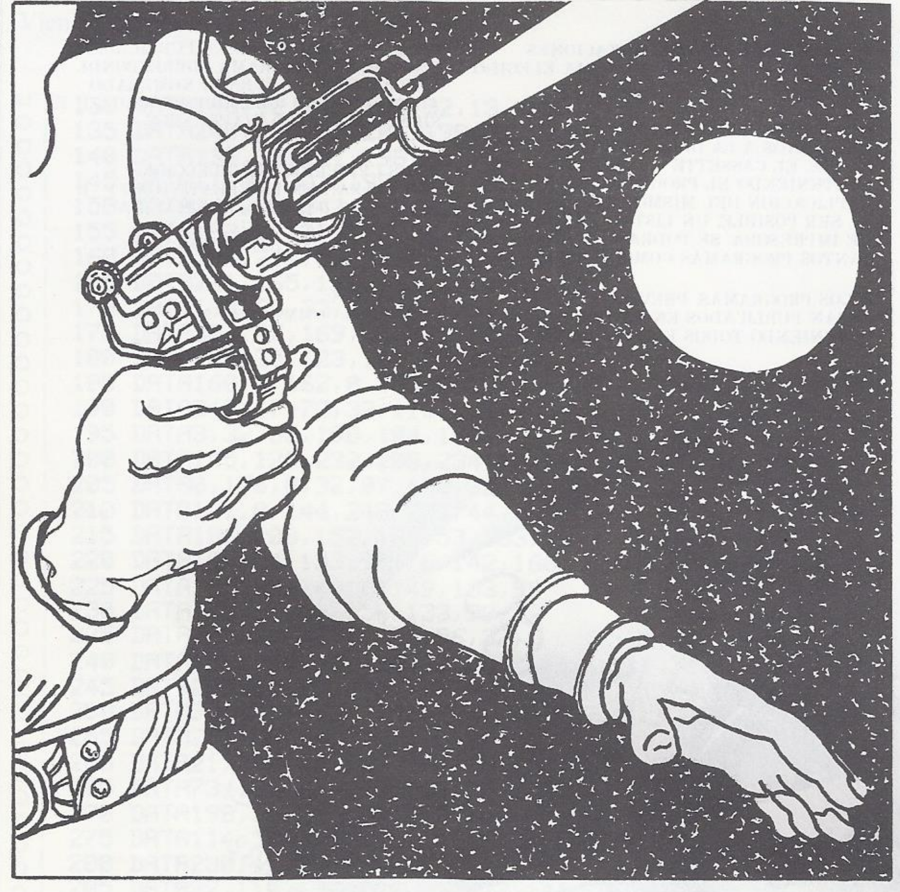
Los programas han de ser inéditos. No deben haberse enviado a otras publicaciones, ni ser copiados de manuales y/o libros, ya sean españoles o extranjeros.



commodore Magazine

C/Bravo Murillo, 377. Madrid - 28002

Programas



Galáctica

en la negrura del espacio exterior a nuestra galaxia, sólo puede verse la mira de nuestro láder y una navecilla invasora que hay que destruir para conseguir puntos. El programa incluye instrucciones al principio sobre cuáles son las teclas que hay que utilizar para desplazar la mira del láser y para disparar. La navecilla enemiga no se está quieta un solo momento, por lo que hay que ser bastante hábil para cazarla. El tiempo es limitado, y al acabar la partida el ordenador indica los puntos conseguidos y escribe una frasecilla comentando si se ha hecho bien o no tan bien. Hay nueve niveles de dificultad, desde el nivel nueve para principiantes al nivel uno para expertos.

VIC 20

```
10 REM *************
20 REM *           *
30 REM * GALACTICA *
40 REM *           *
50 REM *************
100 GOSUB240
110 TI$="000000"
120 POKE36879,26:PRINT"❏"
130 POKE36879,8:A=7932
140 FORB=38400TO38884+21:POKEB,7:NEXT
150 GOSUB 640
160 IFVAL(TI$)>200THEN420
170 IF PEEK(197)=9ANDA>7724THENGOSUB610:A=A-22
180 IFPEEK(197)=26ANDA<8120THENGOSUB610:A=A+22
190 IFPEEK(197)=17ANDA>7703THENGOSUB610:A=A-1
200 IFPEEK(197)=18ANDA<8136THENGOSUB610:A=A+1
210 IFPEEK(197)=41THENGOSUB750
220 GOSUB580:IFU2=1THENPOKEU3,0
```

```
230 GOTO150
240 POKE36879,26:PRINT"■■***GALACTICA**■"
250 PRINT"■DIRIGE TU LASER"
260 PRINT"■    W          A     D     ■       X"
270 PRINT"■■S■  FUEGO"
280 PRINT"■■TIENES 2 MINUTOS "
290 PRINT"■■PULSA UNA TECLA"
300 GETO$:IFO$=""THEN300
310 POKE36879,30:PRINT"■■■****** NIVEL *****■"
320 PRINT"■■DE   1   A   9"
350 GETT$:IFT$=""THEN350
360 T1=VAL(T$)
370 IFT1<1ORT>9THEN350
380 L1=INT(T1)
390 T1=INT(T1*1.3)
400 IFT1=1THENT1=2
410 RETURN
420 POKE36879,26:POKE198,0
440 PRINT"■■NIVEL ";L1
450 PRINT"■■CON ";SC*(10-INT(T1/1.3));"PUNTOS"
460 PRINT
470 SS=SC*(10-INT(T1/1.3))
480 IFSS>15000THEN PRINT"■GENIAL■■■":END
490 IFSS>13000THEN PRINT"■MUY BIEN■■■":END
500 IFSS>12000THENPRINT"■BIEN■■■":END
510 IFSS>5000THENPRINT"■NO ESTA MAL■■":END
520 IFSS>4000THENPRINT"■PUEDE PASAR■■■":END
530 IFSS>3000THENPRINT"■SIGUE PROBANDO■■":END
540 IFSS>2000 THENPRINT"■REGULAR■■■":END
550 IFSS>1000THENPRINT"■FLOJILLO■■■":END
570 END
580 POKEA+2,64:POKEA+3,91:POKEA+4,64:POKEA-2,64:POKEA-3,91:POKEA-4,64
590 POKEA+44,93:POKEA+66,91:POKEA+88,93:POKEA-44,93:POKEA-66,91:POKEA-88,93
600 RETURN
610 POKEA+2,32:POKEA+3,32:POKEA+4,32:POKEA-2,32:POKEA-3,32:POKEA-4,32
620 POKEA+44,32:POKEA+66,32:POKEA+88,32:POKEA-44,32:POKEA-66,32:POKEA-88,32
630 RETURN
640 IFU2<>1THENU3=INT(RND(1)*503)+7680:U2=1:POKEU3,0:RETURN
650 U1=INT(RND(1)*T1)
660 IFU1<>1THENRETURN
670 U4=INT(RND(1)*4)+1
680 IFU4=2THENU4=-1
690 IFU4=3THENU4=22
700 IFU4=4THENU4=-22
710 U5=INT(RND(1)*4)+1
720 IFU4*U5+U3>8164+21THENU4=-22
730 IFU4*U5+U3<7680THENU4=22
740 POKEU3,32:U3=U3+U4*U5:POKEU3,0:RETURN
750 GOSUB880
760 IFPEEK(A)<>0THEN:RETURN
770 H1=A-7680:H2=38400+H1
780 POKEH2,2:POKEH2-1,3:POKEH2+1,3:POKEH2-21,3:POKEH2-22,3
790 POKEH2-23,3:POKEH2+21,3:POKEH2+22,3:POKEH2+23,3:
```

Programas

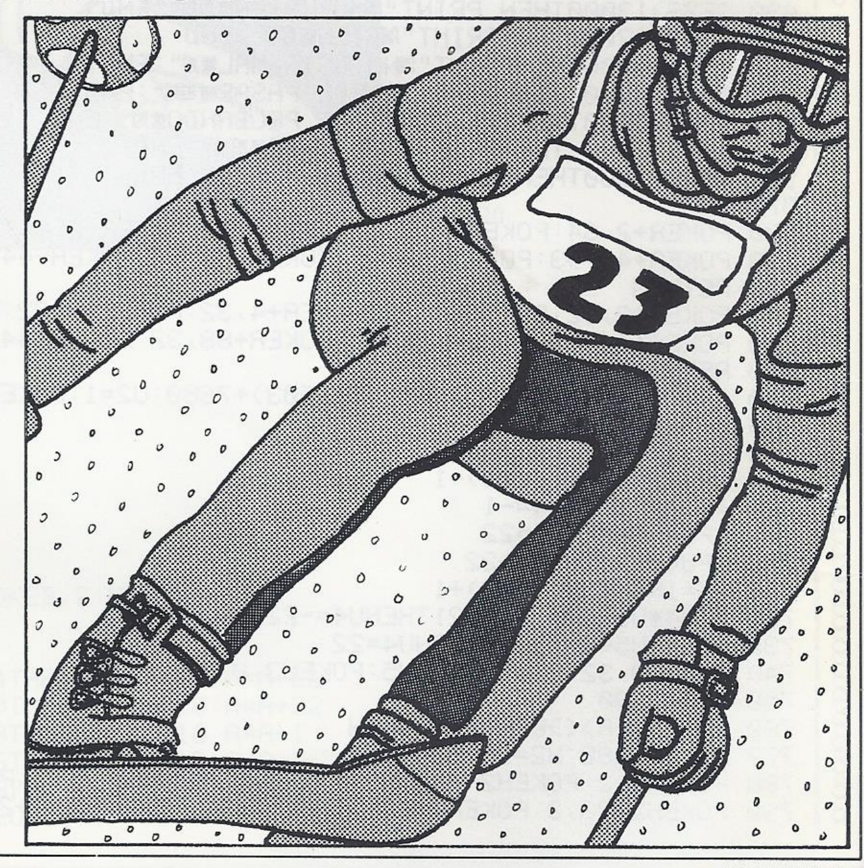
Viene de la página anterior

```
800 POKEA,42:POKEA-1,46:POKEA+1,46:POKEA-22,46:POKEA+22,46
810 POKEA-21,46:POKEA-23,46:POKEA+21,46:POKEA+23,46
820 GOSUB940
830 POKEA-21,32:POKEA-23,32:POKEA+21,32:POKEA+23,32
840 POKEA,32:POKEA-1,32:POKEA+1,32:POKEA-22,32:POKEA+22,32
850 POKEH2,7:POKEH2-1,7:POKEH2+1,7:POKEH2-21,7:POKEH2-22,7
860 POKEH2-23,7:POKEH2+21,7:POKEH2+22,7:POKEH2+23,7:
870 U2=0:SC=SC+100:RETURN
880 POKE36877,0:POKE36878,15
890 FORL=1TO4
900 FORM=250TO240STEP-1:POKE36876,M:NEXT
910 FORM=240TO250:POKE36876,M:NEXT
920 POKE36876,0:NEXT
930 POKE36878,0:RETURN
940 POKE36877,220
950 FORL=15TO0STEP-.05
960 POKE36878,L
970 FORM=1TO1:NEXT
980 NEXT:RETURN
```

PROGRAMA commodore Magazine COMPROBADO

Slalon

CBM-64



Aunque la pendiente es bastante pronunciada, la nieve está en muy buenas condiciones por lo que no debes tener problemas para llegar a la meta. Este juego para el C-64 consiste en batir todos los récords del Slalom Gigante. Lo que ocurre es que en lugar de puertas y banderas, las marcas de pista son unos impresionantes pinos, que es necesario evitar. Además de los pinos, hay de vez en cuando alguna que otra placa de hielo que puede dar con el esquiador en el suelo al menor descuido. Para llegar a la meta esquivando todos los obstáculos hay que dirigir al esquiador mediante las teclas "Z" (izquierda) y "M" (derecha). Además el movimiento continúa en la misma dirección mientras no se pulse la tecla correspondiente para cambiarlo.

A continuación damos una lista de las variables utilizadas por el programa:

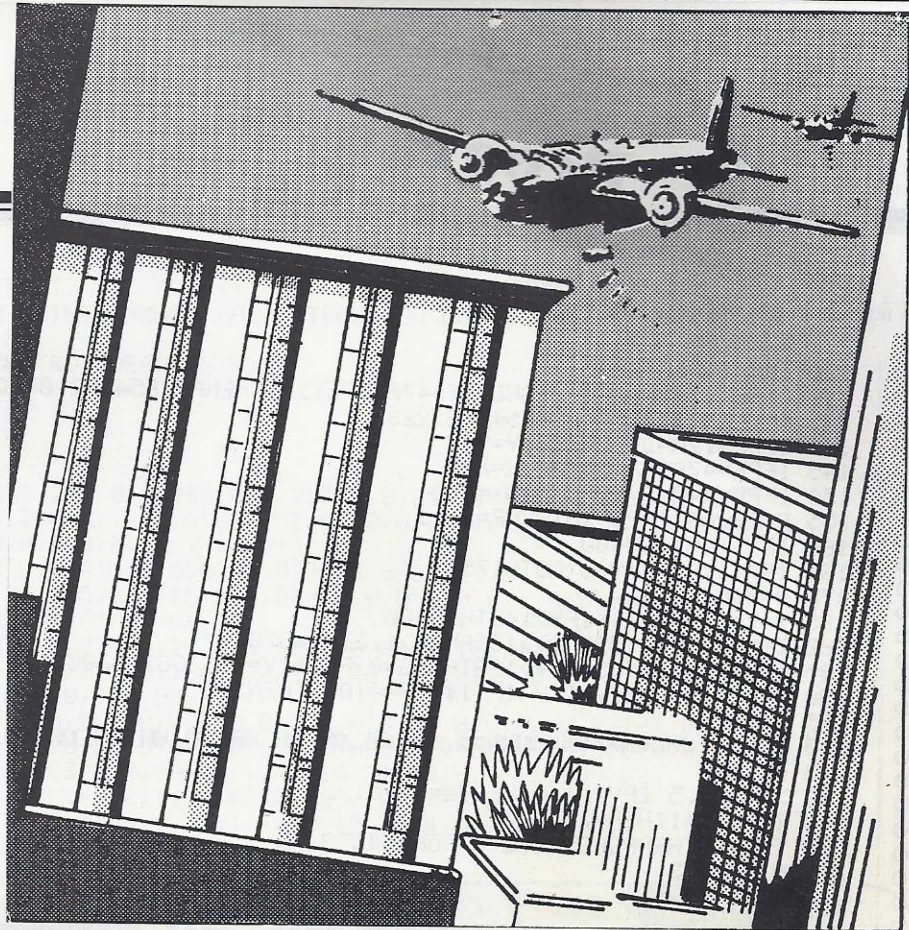
D=Dirección del esquiador.
O=Izquierda.
1=recto.
2=derecha.
V=Dirección del chip de video.
SL=Porcentaje de pista recorrida.
T=No. de pinos.
X1=Posición de pino.
X=Posición horizontal del esquiador.
S=Puntuación.

```
10 REM **********
15 REM *        *
20 REM * SLALOM *
25 REM *        *
30 REM **********
35 REM
40 POKE55,255:POKE56,47:POKE53281,15:POKE53280,15:PRINT"▯"
45 D=1:X=140:V=53248:SL=0:S=0:POKEV+21,1
50 POKE2040,193:POKEV+39,2:FORI=0TO318:READQ:POKE12288+I,Q:NEXT
55 POKE54278,0:POKE54276,0:POKE54273,0:POKE54272,0
60 POKE 54296,10:POKE54278,240:POKE54276,129
65 FORI=12800TO12871:READA:POKEI,A:NEXT:POKEV+24,28:FORI=12544TO12551
70 POKEI,0:NEXT:POKEV,X:POKEV+1,60:Z=PEEK(V+31)
75 S=S+1:SL=SL+1:PRINT"▒▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯▯":IF SL=220THEN340
80 POKE54273,0:PE=PEEK(197):IFSL=180THENGOSUB275
85 IF PE=12ANDD>0THEND=D-1:POKE2040,192+D:POKE54273,100
90 IF PE=36ANDD<2THEND=D+1:POKE2040,192+D:POKE54273,100
95 IFX<45THEND=1:X=45:POKE2040,192+D:GOTO110
100 IFX>239THEND=1:X=239:POKE2040,192+D:GOTO110
105 X=X+(D-1)*3
110 POKEV,X
115 IFPEEK(V+31)=1THEN165
120 IFRND(1)<.7ANDT=0ANDSL<175THENT=1:X1=INT(RND(1)*30)
125 IF PE=12ORPE=36THENPOKE54273,0
130 IF RND(1)<.1ANDT=0ANDSL<175THENPRINTTAB(INT(RND(1)*30));"■|▤AAAAAAA■|"
135 IF RND(1)<.1ANDT=0ANDSL<175THENPRINTTAB(INT(RND(1)*30));"▩♠♠♠♠":GOTO75
140 IFT=0THENPRINT:GOTO75
145 IFT=1THENPRINTTAB(X1);"▮▮▛":T=2:GOTO75
150 IFT=2THENPRINTTAB(X1)"▮▙─▔|":T=3:GOTO75
155 IFT=3THENPRINTTAB(X1)"▮▙__":T=4:GOTO75
160 IFT=4THENPRINTTAB(X1)"▮ |":T=0:GOTO75
165 IFSL>197THEN285
170 IFPEEK(1104+(X-20)/8)=1OR PEEK(1405+(X-20)/8)=1THENGOTO120
175 IFPEEK(1106+(X-20)/8)=1THENGOTO120
180 IFPEEK(1144+(X-20)/8)=1ORPEEK(1145+(X-20)/8)=1THENGOTO260
185 IFPEEK(1146+(X-20)/8)=1THENGOTO260
190 IFPEEK(1064+(X-20)/8)=1ORPEEK(1065+(X-20)/8)=1THENGOTO120
195 IFPEEK(1066+(X-20)/8)=1THENGOTO120
200 IFPEEK(1024+(X-20)/8)=1ORPEEK(1025+(X-20)/8)=1THENGOTO120
205 IFPEEK(1026+(X-20)/8)=1THENGOTO260
210 POKE54296,15
215 POKE2040,196:FORI=60TO170:POKE54273,175-I:POKEV+1,I:NEXT
220 POKE 54273,1
225 POKE 2040,195
230 FORI=1TO700:NEXT:POKE54273,0
235 POKEV,0:POKEV+1,0:POKE53280,0:POKE53281,0:POKEV+24,20
240 PRINT"▯▮MALA SUERTE TE HAS ESTRELLADO!"
245 PRINT"▯▯▯▯◣HAS COMPLETADO EL▤"         INT(SL/2);"% DE LA"
250 PRINT"CARRERA.▩PUNTUACION "         INT(S/4)"PUNTOS."
255 GOTO315
260 S=S*2:POKE54296,15:POKE54276,0:POKE54276,33:FORI=20TO50:POKE54273,I:NEXT
265 POKE54296,15:POKE54276,0:POKE54276,129:POKE54273,0
270 GOTO120
275 PRINT"▯▮▮▮▮▮▮▮▮▮▮▮▮▮▮ -▜⊢▜⊢──────────▒ ▔▮▛▮▛"
280 RETURN
285 POKE54273,10:POKE2040,193:FORI=1TO5:FORJ=1TO20STEP1:FORZ=1TO20:NEXT
290 POKEV+1,50+J:NEXTJ:FORJ=20TO1STEP-1:FORZ=1TO20:NEXT:POKEV+1,50+J:NEXTJ,I
295 FORI=1TO240STEP5:POKE54273,I:NEXT:POKE54273,0
300 POKEV,0:POKEV+1,0:POKE53280,0:POKE53281,0:POKEV+24,20
```

Programas

Viene de la página anterior

```
305 PRINT"■■HAS TERMINADO LA CARRERA!"
310 PRINT"■■PUNTUACION ▶";INT(S/4);"% PUNTOS."
315 PRINT"■■■QUIERES VOLVER A JUGAR?S/N"
320 GETA$:IFA$="S"THEN RUN
325 IFA$<>"N"THEN320
330 PRINT"■■■ADIOS"
335 END
340 FORI=1TO240:POKE54273,I:NEXT:POKE54273,0:POKE53280,0:POKE53281,0
345 POKEV,0:POKEV+1,0:POKEV+24,20:PRINT"■■HA HA!"
350 PRINT"■▶TE PASASTE EL POSTE DE LA META!"
355 PRINT"■■MALA SUERTE."
360 PRINT"■■PUNTUACION ";S;" PUNTOS."
365 GOTO315
370 DATA0,56,0,0,56,0,0,56,0,0,16,0,0,124,0,0,187,0,1,57,0,1,57,0,0,185
375 DATA0,0,165,0,0,36,0,0,68,0,0,66,0,0,34,0,0,35,0,0,50,0,0,36,0,0,72,0,
380 DATA0,144,0,1,32,0,2,64,0,99 (ESQUIADOR)
385 DATA0,56,0,0,56,0,0,56,0,0,16,0,0,124,0,0,187,0,1,57,0,1,57,0,0,185
390 DATA0,0,165,0,0,36,0,0,68,0,0,66,0,0,42,0,0,38,0,0,50,0,0,33,0,0,64,128
395 DATA0,128,64,1,0,0,2,0,0,99
400 DATA0,56,0,0,56,0,0,56,0,0,16,0,0,124,0,0,187,0,1,57,0,1,57,0,0,185
405 DATA0,0,165,0,0,36,0,0,68,0,0,66,0,0,42,0,0,38,0,0,98,0,0,33,0,0,16,128
410 DATA0,8,64,0,4,32,0,2,16,99
415 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
420 DATA0,0,0,16,0,8,8,0,16,4,0,32,2,0,64,1,0,128,0,129,0,2,66,64,
425 DATA0,0,0,0,255,0,99  DIREC. 12543
430 DATA2,0,0,1,0,0,0,128,64,
435 DATA0,64,128,0,33,0,0,50,0,0,38,0,0,42,0,0,66,0,0,68,0,0,36,0,0,165,0
440 DATA0,185,0,1,57,0,1,57,0,0,187,0,0,124,0,0,16,0,0,56,0,0,56,0,0,56,0,
445 DATA255,255,255,255,255,255,255,255
450 DATA0,0,0,59,255,231,0,0,128,192,224,240,128,128,128,128,3,3,3,3,3,3,3,3,
455 DATA 192,192,192,192,192,192,192,192
460 DATA0,0,24,24,60,126,126,255,3,3,7,15,31,63,63,127,192,192,224,240,240
465 DATA248,252,252,60,60,60,60,60,60,60,60
```



Harrier

Con este juego para el C-64 echamos sobre tus espaldas una gran responsabilidad. Estás a cargo de un avión Harrier transportado por un portaviones y tu misión es destruir un submarino enemigo que se dirige hacia las costas de tu país. Sólo dispones de un minuto para completar tu misión, y de todas las cargas de profundidad que necesites. Ten buen cuidado con las cargas de profundidad para no detonar una mina nuclear puesta por tu propio ejército y que anda por los alrededores. En la parte derecha de la pantalla aparece un indicador de combustible. Si se te acaba cuando estés en el aire te estrellarás sin remedio, pero tienes siempre la posibilidad, antes de que esto ocurra, de volver al portaviones y aterrizar en él, eso sí muy muy suavemente.

Las teclas para controlar el avión son:

Z = Arriba
C = Abajo
= Izquierda
= Derecha
Espacio = Lanzador de cargas de profundidad.

Lo que sigue es una lista de las variables utilizadas por el programa y de su función:

Y, Y = Posición horizontal y vertical del sprite 0 (harrier).
X1, Y1 = Posición horizontal y vertical del sprite 1 (submarino).
X2, Y2 = Posición horizontal y vertical del sprite 3 (carga de profundidad).
CO = Chip de sonido.
V = Chip de video.

CBM-44

```
10 REM ************************
15 REM *       HARRIER        *
20 REM *                      *
25 REM * COMMODORE MAGAZINE *
30 REM ************************
35 POKE55,255:POKE56,47:PRINT"♥":V=53248:CO=54272:POKE53281,0:POKE53280,2:F=10
40 FORI=55816TO56296:POKEI,6:POKEI-CO,160:NEXT:G=0
45 POKEV+21,63:POKE2040,192:POKE2041,193:POKE2042,194:POKE2043,195:POKEV+44,7
50 POKE2044,195:POKE2045,196:POKEV+39,1:POKEV+40,14:POKEV+41,0:POKEV+42,21
55 POKEV+43,7:POKEV+29,18:POKEV+23,16:FORI=0TO318:READQ:POKE12288+I,Q:NEXT:X=26
60 Y=133:X1=29:Y1=160+INT(RND(1)*60):POKEV,X:POKEV+1,Y:D=1:POKEV+2,X1:POKEV+3,Y1
65 :FORJ=0TO24:FORI=55330TO55335:POKEI+J*40,2:POKEI+J*40-CO,160:NEXTI,J
70 POKEV+5,160+INT(RND(1)*60):POKEV+4,70+INT(RND(1)*180)
75 PRINT"▓▓▓▓▓▓▓▓▓▓▓▓▓▓ ▀▓▓▓      ▄▓":POKEV+8,210:POKEV+9,55:GOSUB200
80 POKE54296,1:POKE54278,240:POKE54276,129:TI$="000000"
85 PRINT"▓▓▓█████████▓TIEMPO"
90 PRINT"▓▓▓▓█████████▓";RIGHT$(TI$,2):POKE55652+INT(F)*40,4
95 X1=X1+D*2:IFX1>240THEN D=INT(RND(1)*3-3):Y1=160+INT(RND(1)*60)
100 IFX1<28 THEND=INT(RND(1)*3+1):Y1=160+INT(RND(1)*60)
105 POKEV+2,X1:POKEV+3,Y1
110 PE=PEEK(197)
115 IFPE=60ANDX>60ANDX2=0THENX2=X:Y2=Y+8:G1=0:F=F-.1:POKE53273,200
120 IFX2>0THENGOSUB210
```

Programas

Viene de la página anterior

```
125 IFF<1THEN160
130 IFPE<>12ANDPE<>20ANDPE<>47ANDPE<>44THENPOKE54272,0:POKE54273,0:GOTO175
135 POKE54273,108:POKE54272,223
140 IFPE=12ANDY>50THENY=Y-2
145 IFPE=47ANDX>28THENX=X-4
150 IFPE=44ANDX<230THENX=X+6
155 F=F-.06:IFPE=20THENF=F-.1
160 IFY>140THEN400
165 IFF<1THENY=Y+G:GOTO175
170 G=0:GOTO185
175 Y=Y+G:G=G+.1:IFY>140THEN400
180 IFX>25ANDX<59ANDY>133ANDG>1.3THEN370
185 IFX>25ANDX<59ANDY>133THENG=0:F=10:Y=133:GOSUB200
190 POKEV,X:POKEV+1,Y:IFTI$="000100"THEN335
195 GOTO90
200 PRINT"SQQQQQQQQQQIIIIIEEL_QL_QL_QL_QL_QL_QL_QL_QL_QL"
205 RETURN
210 X2=X2+.5:IFY2>140THENY2=Y2+G1/2
215 IFY2<141THENY2=Y2+G1
220 IFG1=0THENPOKE54272,0:POKE54273,40
225 G1=G1+.3
230 POKEV+7,Y2
235 POKEV+6,X2
240 IFY2>230THENY2=0:X2=0:POKEV+7,0:G1=0
245 IFPEEK(V+30)=10THEN260
250 IFPEEK(V+30)=12THEN295
255 RETURN
260 POKEV+2,0:POKEV+3,0:POKEV+10,X1:POKEV+11,Y1:POKE54296,15
265 FORI=70TO10STEP-.3:POKE54273,I:NEXT:POKE54273,0:POKE54272,0
270 FORI=1TO11:POKEV+I,0:NEXT
275 POKEV+10,0:POKEV+11,0:PRINT"JQQQ=HAS DESTRUIDO EL SUBMARINO!!!!!!!!!"
280 PRINT"QQQBRAVO!!!"
285 PRINT"QQTIEMPO ";RIGHT$(TI$,2);" SEGUNDOS"
290 GOTO425
295 FORI=1TO11:POKEV+I,0:NEXT:POKEV+29,32:POKEV+23,32
300 POKEV+2,0:POKEV+3,0:POKEV+10,X2:POKEV+11,Y2:POKE54296,15:POKE53281,8
305 FORI=90TO0STEP-.2:POKE54273,I:POKE53280,I/10:NEXT:POKE54273,0:POKE54272,0
310 POKE53280,0:POKE53281,0:POKEV+29,0:POKEV+23,0
315 POKEV+10,0:POKEV+11,0:PRINT"JQQQ=HAS ALCANZADO LA MINA NUCLEAR!!!!!!!!!"
320 PRINT"QQTEN MAS CUIDADO!!!"
325 PRINT"QQHAS RESISTIDO  ";RIGHT$(TI$,2);" SEGUNDOS."
330 GOTO425
335 POKE54296,15:POKE54276,0:POKE54276,33
340 FORI=255TO0STEP-2:POKE54273,I:NEXT:POKE54273,0:POKE54272,0
345 FORI=0TO11:POKEV+I,0:NEXT:POKE53280,0:POKE53281,0
350 PRINT"JQQ=SE ACABO EL TIEMPO!"
355 PRINT"JQQEL SUBMARINO VA A DESTRUIR TU PAIS!!!!!!"
360 PRINT"QQQLA QUE HAS ORGANIZADO!!"
365 GOTO425
370 POKEV,0:POKEV+1,0:POKEV+10,X:POKEV+11,Y:POKE54296,15
375 FORI=70TO20STEP-.2:POKE54273,I:NEXT:POKE54273,0:POKE54272,0
380 FORI=1TO11:POKEV+I,0:NEXT
385 POKEV+10,0:POKEV+11,0:PRINT"JQQ=TE HAS ESTRELLADO EN TU PROPIO BARCO!!"
390 PRINT"QQMALA SUERTE!"
395 GOTO425
400 POKEV,0:POKEV+1,0:POKEV+10,X:POKEV+11,Y:POKE54296,15
405 FORI=80TO30STEP-.2:POKE54273,I:NEXT:POKE54273,0:POKE54272,0
```

PROGRAMA commodore COMPROBADO

```
410 FORI=1TO11:POKEV+I,0:NEXT
415 POKEV+10,0:POKEV+11,0:PRINT"TE HAS ESTRELLADO EN EL MAR!!!!!"
420 PRINT"MALA SUERTE."
425 PRINT"OTRO JUEGO (S/N)?"
430 GETA$:IFA$="S"THENRUN
435 IFA$<>"N"THEN430
440 PRINT"ADIOS"
445 END
450 DATA0,0,0,0,0,0,192,0,0,192,0,0,240,0,0,252,3,0,255,255,192,255,255,252
455 DATA63,255,255,63,255,240,31,240,0,63,252,0,63,0,0,0,0,0,0,0,0,0,0,0,0,0,
460 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,99
465 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,56,0,0,56,0,0,124,0,3,255,192,127,255
470 DATA252,255,255,255,255,255,252,15,255,224,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
475 DATA0,0,0,0,0,0,0,0,99
480 DATA0,0,0,0,0,0,0,0,0,0,0,4,8,16,2,8,32,1,8,64,0,136,128,0,127,0,0,127,0
485 DATA7,255,240,0,127,0,0,127,0,0,136,128,1,8,64,2,8,32,4,8
490 DATA16,0,0,0,0,0,0,0,0,0,0,0,0,99
495 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,28,0,0,62,0,0,127,0,0
500 DATA127,0,0,127,0,0,62,0,0,28,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
505 DATA99,128,0,32,18,1,8,233,0,4,99,63,1,48,84,32,19,8,1,89,109,22
510 DATA8,12,38,0,52,149,128,243,211,99,63,0,128,72,1,19,8,0,89,2,22
515 DATA99,0,2,18,76,149,2,64,1,99,4,7,91,16,0,19,8,54,1,109,22
```



Bajo esta denominación se acostumbra a considerar incluidos un determinado tipo de periférico de reducido precio, pero como se verá, de elevadas prestaciones que permite implantar un diálogo fuertemente interactivo y eficaz entre el operador y el sistema vía el T.R.C. (Tubo de Rayos Catódicos) del mismo.

FILOSOFIA DE OPERACION

La primera precisión que se debe realizar es la no confusión del Lápiz óptico con el lector óptico de caracteres o con el lector de código de barras. En efecto, mientras que estos dos últimos se utilizan para decodificar la información digital (caracteres OCR y barras) contenida en una imagen sobre papel u otro soporte, aquel tan solo puede proporcionar una señal lógica de la presencia o ausencia de un nivel luminoso adecuado sobre la pantalla TRC del sistema.

Ambos procesos se diferencian fundamentalmente como puede comprobarse en la fig. 1.

En la que se observa como el lápiz óptico se limita a proporcionar una señal de interrupción al sistema; señal que decodifica en términos del *timing* respecto a las señales de exploración de la pantalla, permite identificar con precisión las coordenadas X-Y (fila, columna) del punto luminoso sobre la misma.

APLICACIONES DEL LAPIZ OPTICO

Las principales aplicaciones de los lápices ópticos se encuentran en aquellos campos en que interesa aumentar la interacción del operador con el sistema, simplificar el trabajo del mismo o bien, aumentar la velocidad de introducción e interpretación de datos durante el proceso. Huyendo de un análisis exhaustivo, que está fuera de los alcances de este trabajo, se podrán considerar al menos las siguientes posibilidades:

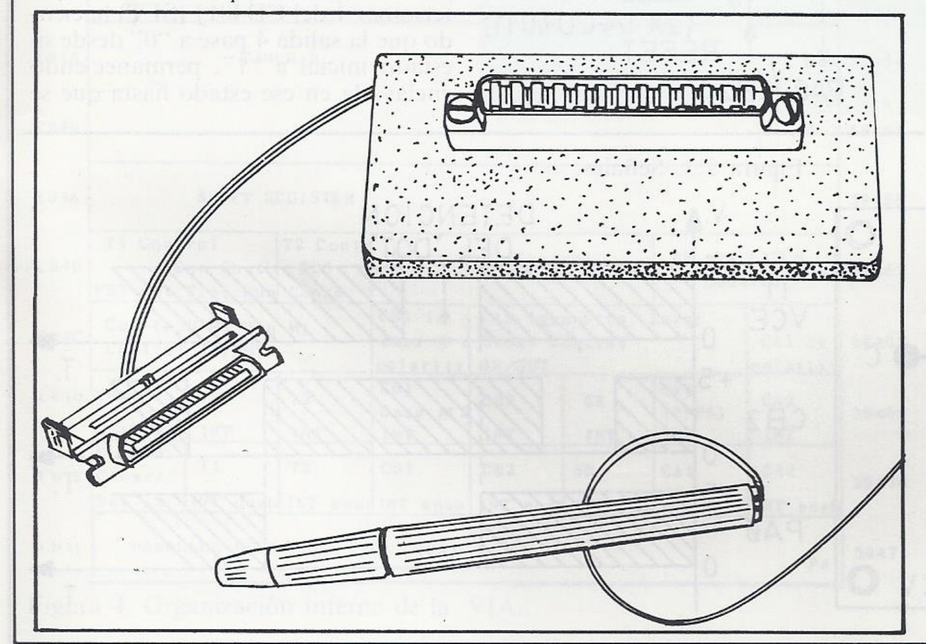
a) *Trabajos por operadores no experimentados:* En determinadas aplicaciones docentes (y algunas de gestión) el lápiz óptico puede facilitar el trabajo de selección de datos u opciones sobre muestras en la pantalla, a operadores que posean la simple y modesta condición de poder leer los textos sobre la misma.

b) *Trabajos de selección rápida de opciones en ambientes "ruidosos" o de alto nivel de confusión:* En estos ambientes, la efectividad del operador, así como su rendimiento, decrece rápidamente con el tiempo, la utilización del lápiz óptico. Permite al simplificar las operaciones, que éstas

0 0 1 0 1
ZONA SUP.
ZONA INF.
0 0 0 0 1
C / OCR
Primer dígito
Dos de cinco
Ejemplo de código de barras
Principio 0 0 2 2 1 4 9 Final
C / BARRAS
INTERFACE
DATOS
SISTEMA
LECTORES
(-OCR)
(-RBC)
SEÑAL DE INTERRUPCION
INTERFACE
LAPIZ OPTICO (L.O.)
Y
X
TRC

Figura 1.

Figura 3. Prototipo.



lleguen a producirse en forma mecánica sobre los códigos existentes en la pantalla, al tiempo que se toman los datos (por teléfono, por ejemplo) elevando la fiabilidad del sistema.

c) *Trabajos de diseño asistido por ordenador:* Es el campo más específico de aplicaciones de los lápices ópticos. A este respecto, cualquier programador que haya intentado enfrentarse a un simple diseño geométrico sobre la pantalla, contando únicamente con sentencias PRINT, POKE o equivalentes, puede evaluar la conveniencia de poder utilizar una herramienta que "recoja" los caracteres y colores de una muestra colocada sobre una zona de la pantalla y las traslade a otra en la que poco a poco y con las rectificaciones necesarias, se construye el diseño definitivo. Diseño, que puede ser luego archivado sobre disco o cinta bien con una utilitario "que salve" el *buffer* de pantalla o bien en un fichero con

registros alfanuméricos (uno por línea de la misma).

HARDWARE Y SOFTWARE ASOCIADOS

El *hardware* básico con que deben contar estos periféricos está íntimamente relacionado con el sistema de captación y consta, en esencia, de un elemento sensible a la luz (generalmente un fototransistor o un fotodiodo), un acondicionador de señal y en algunos casos como se verá, una báscula o *"lacth"*, que permite memorizar el pulso recibido, así como un pulsador que pone ON y OFF la función del periférico.

El *software* necesario depende, por supuesto, de la función final a realizar, pero de las posibilidades de interpretar la interrupción que incluya el sistema operativo.

REALIZACION Y PROGRAMACION DEL PERIFERICO

A continuación se describen realizaciones distintas del lápiz óptico, sobre distintos microordenadores **(CBM 4000-8000 y VIC-20)** con los que el autor ha tenido ocasión de experimentar.

LAPIZ TOTALMENTE CONTROLADO POR SOFTWARE

En sistemas tales como el **CMB 4000 y 8000** (también es válido para los viejos **3000** y **2000**) y, en general, para cualquier equipo que no cuenta con prestaciones para lápiz óptico, implementadas en *hardware* y *software* de origen (como las del **VIC-20**), puede montarse fácilmente un simple circuito como el de la figura 2.

El funcionamiento del mismo a nivel circuital es bastante sencillo:

Las dos puertas NAND forman una báscula elemental RS, que es activada por el paso del transistor T1 del estado de corte al de saturación, al recibir un nivel adecuado de excitación luminosa sobre su base, lo que provoca que se aplique un "cero" al terminar 1 del CD4011 (SET) haciendo que la salida 4 pase a "0" desde su estado inicial a "1", permaneciendo enclavada en ese estado hasta que se

Figura 2.1. Circuito.

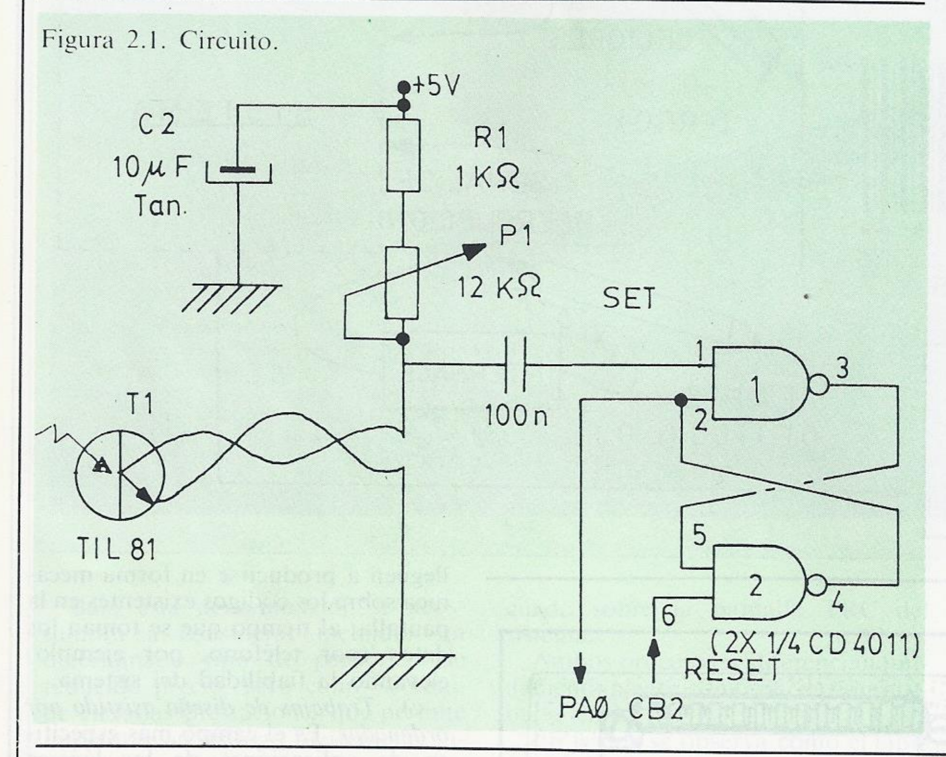


Figura 2.2. Placa de circuito impreso.

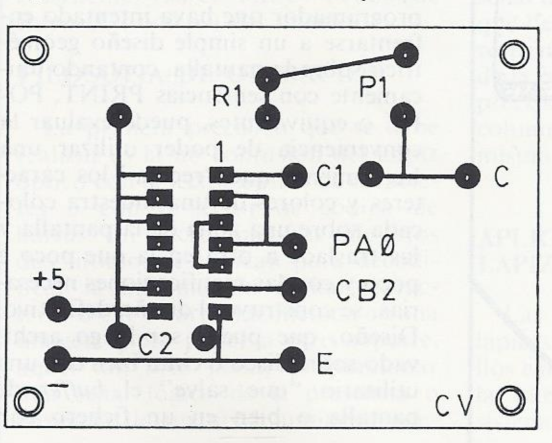
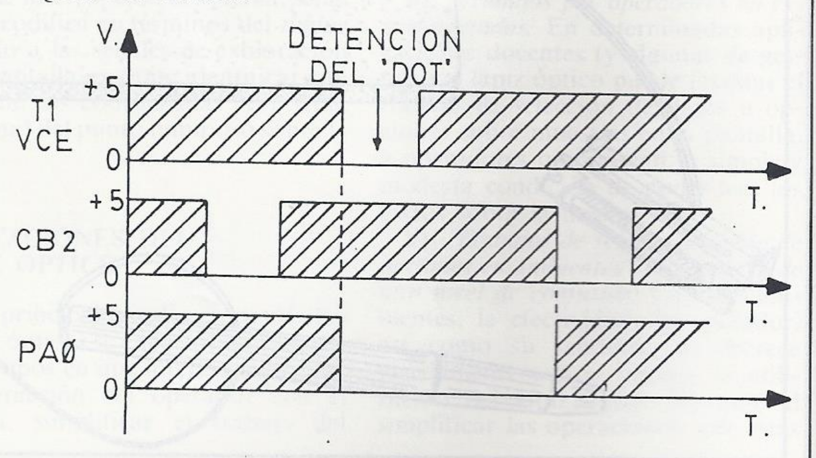


Figura 2.3. Señales.



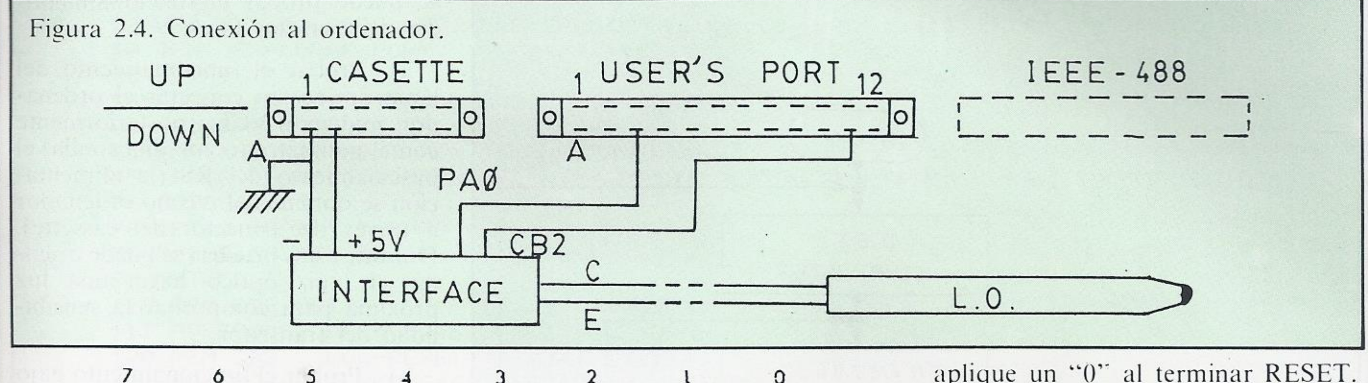


Figura 2.4. Conexión al ordenador.

	7	6	5	4	3	2	1	0	
E840	DAV in	NRFD in	RETRACE in	CASS # 2 MOTOR	CASSETTE OUTPUT	ATN out	NFRD out	NDAC in PB	59456
E841	ORA (CONTROLS HANDSHAKES)								59457
E842	DIRECTION REGISTER B (FOR E840)								59458
E843	DIRECTION REGISTER A (FOR E84F) (P.U.P.)								59459
E844	TIMER 1							L	59460
E845								H	59461
E846	TIMER 1							L	59462
E847	LATCH							H	59463
E848	TIMER 2							L	59464
E849								H	59465
E84A	SHIFT REGISTER								59466
E84B	T1 Control One-Shot PB7 out, Free Run		T2 Contr PB6 Sense	Shift Reg. Control			PB, PA LATCH Control		59467
E84C	CB2 (P.U.P. Pin M) Control IN/OUT			CB1 in cass. # 2 polarity	CA2 (graphics, Lower Case) Control IN/OUT			CA1 in polarity	59468
E84D	IRQ STATUS	T1 INT	T2 INT	CB1 Cass # 2 INT	CB2 INT	SR INT	CA1 (PUPB) INT	CA2 INT	59469
E84E	Enable Clear/ Set	T1 INT enab	T2 INT enab	CB1 INT enab	CB2 INT enab	SR INT enab	CA1 INT enab	CA2 INT enab	59470
E84F	PARALLEL USER PORT I/O (PA)							PA	59471

Figura 4. Organización interna de la VIA.

aplique un "0" al terminar RESET.

El conjunto puede encerrarse en una caja como la de la figura 3 y es inevitable utilizar un tubito de plástico o de metal para encerrar el fototransistor con su lente debidamente protegida de reflexiones laterales.

Para un correcto funcionamiento, la secuencia de las señales debe ser como la indicada en la figura 2.3.

Es fácil obtener las señales necesarias para el RESET a partir del CB2 de la 6522 VIA del *port* del usuario, aunque para conseguir utilizar el mismo, es necesario programarlo previamente. Para ello es una buena ayuda el contar con la descripción de la organización interna de la VIA que puede verse en la figura 4.

Como se ve en la misma, el CB2 ocupa los bits 7, 6 y 5 de la dirección 59468. Por ello y enmascarando adecuadamente para no afectar al resto de los bits, se pueden utilizar las siguientes sentencias para controlar esta línea de E/S desde el Basic:

```
POKE 59467, PEEK (59467)
AND 277 (PONER CB2 OUT)

POKE 59468, PEEK (59468)
AND 31 OR 192 (CB2 BAJO)

POKE 59468, PEEK (59468)
AND 31 OR 224 (CB2 ALTO)
```

De otra parte, el estado de la salida 4 PA0 puede ser leído con facilidad, sin más que consultar la DDR de la VIA en 59471 toda vez que si no se altera la ORA (59457) todos los bits PA0 a PA7 se encuentran definidos como entradas (y normalmente a "1").

Una vez aclarados estos conceptos,

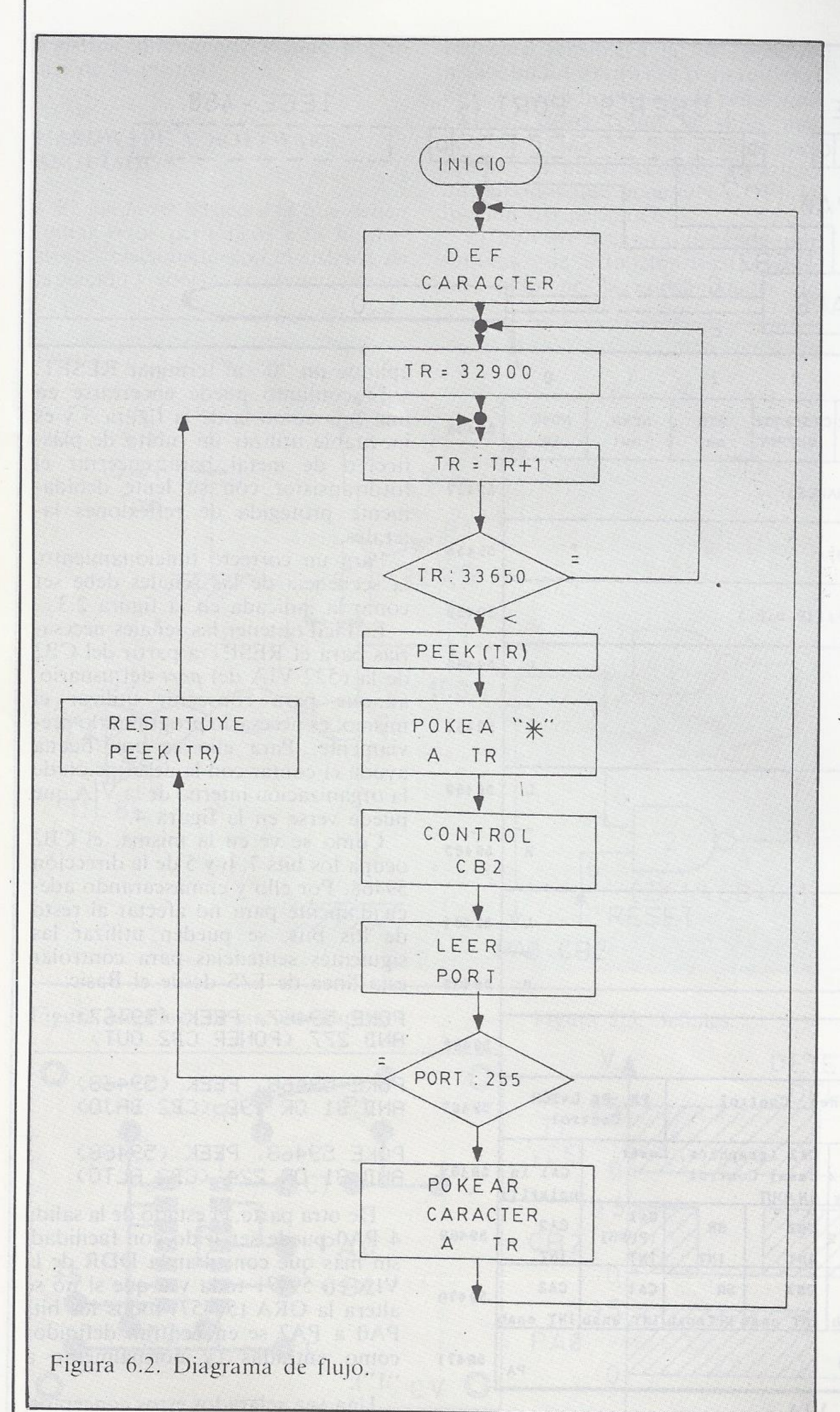


Figura 6.2. Diagrama de flujo.

se puede probar el funcionamiento del lápiz en dos fases:

a) Probar el funcionamiento del lápiz óptico sin conectar el ordenador, midiendo VCE y posteriormente con el polímetro (o con una sonda) el basculamiento del RS (la alimentación se obtiene del mismo ordenador a través del conector del cassette). Durante estas pruebas se puede orientar el lápiz óptico hacia una luz próxima para comprobar la sensibilidad del transistor.

b) Probar el funcionamiento bajo control del ordenador. Para ello se debe entrar el programa de la figura 5.

Este programa visualiza el contenido del *port* que será 255 (= 11111111 binario) si no se excita la base de T1.

Las principales sentencias se explican a continuación:

55: Realiza la secuencia CB2 OUT/BAJO/ALTO para preparar los estados del RS.

60: Se lee el *port* y se imprime.

75: Si la DDR es 255, se ha detectado un pulso y el programa salta al lazo de detención en la 78.

79: Si se detecta un pulso y posteriormente se salió del lazo en 78; pulsando una tecla cualquiera se realiza CB2 BAJO (RESERT).

111/120/300: Subrutinas OUT/BAJO/ALTO.

Al hacer RUN al programa con el lápiz situado sobre la parte derecha de la pantalla, aparecerá una columna de "255" en el lado izquierdo de la misma. Al llevar el lápiz sobre ésta, el valor visualizado debe cambiar a "254" e interrumpirse la ejecución hasta que se pulse una tecla cualquiera. Incidentalmente puede ser necesario ajustar P1 (o el brillo, o ambos) hasta obtener un funcionamiento fiable.

Pasada esta prueba se puede proceder a experimentar con un program de aplicación, como es el caso del "situador de letras" del programa de la misma. Al llevar el lápiz sobre ésta, el valor visualizado debe cambiar a

```
10 REM BY  J. ROCA DTO. ELECTRONICA E.U.P.C.-U.M. *********
40 REM  EXPLORAR PORT        ***********
50 PRINT" "
55 GOSUB110:GOSUB120:GOSUB300:REM CONTROL L.O.====
60 PRINTPEEK(59471)," ":REM LEER DDR =
62 FORI=1TO100:NEXT:REM TEMPORIZAR ===
75 IFPEEK(59471)<>255THENPRINT"":PRINT"    PULSE  SPACE   PARA RESET":GOTO79
76 GOTO80
78 GETA$:IFA$=""THEN78
79 GOSUB120:REM RESET         ============
80 GOTO50
110 REM CB2 OUT              ************
111 POKE59467,PEEK(59467)AND277:RETURN
120 REM CB2   LOW-RESET      ************
121 POKE59468,PEEK(59468)AND31OR192:RETURN
300 REM CB2   HIG            ************
301 POKE59468,PEEK(59468)AND31OR224:RETURN
READY.
```

Figura 5. Prueba del lápiz óptico.

```
10 REM BY  J. ROCA DTO. ELECTRONICA E.U.P.C.-U.M. *********
19 REM DEMO SITUADOR DE LETRAS *******
20 GOTO1000
110 REM CB2 OUT              ************
111 POKE59467,PEEK(59467)AND277:RETURN
120 REM CB2   LOW-RESET      ************
121 POKE59468,PEEK(59468)AND31OR192:RETURN
300 REM CB2   HIG            ************
301 POKE59468,PEEK(59468)AND31OR224:RETURN
1000 A=32900:PRINT" "
1015 PRINT"  CARACTER ?"
1016 GETA$:IFA$=""THEN1016
1017 C=ASC(A$):IFC>64THENC=C-64:REM CONVERSION ASC-POKE
1018 GOSUB110:GOSUB120:GOSUB300:REM CONTROL L.O.
1020 FORI=1TO750:REM EXPLORAR ========
1030 B=PEEK(A+I):POKEA+I,160
1035 FORR=0TO15:NEXT
1038 IFPEEK(59471)<>255THENPOKEA+I,C:I=751:GOTO1050:REM DETECCI
1040 POKEA+I,B
1050 NEXT
1060 GOTO1015
```

Figura 6.1. Listado.

"254" e interrumpirse la ejecución hasta que se pulse una tecla cualquiera. Incidentalmente puede ser necesario ajustar P1 (o el brillo, o ambos) hasta obtener un funcionamiento fiable.

Pasada esta prueba se puede proceder a experimentar con un programa de aplicación, como es el caso del "situador de letras" del programa de la figura 6.

Joaquín Roca Dorda
Cátedra de Electrónica.
E.U.P. de Cartagena.

(Continuará)

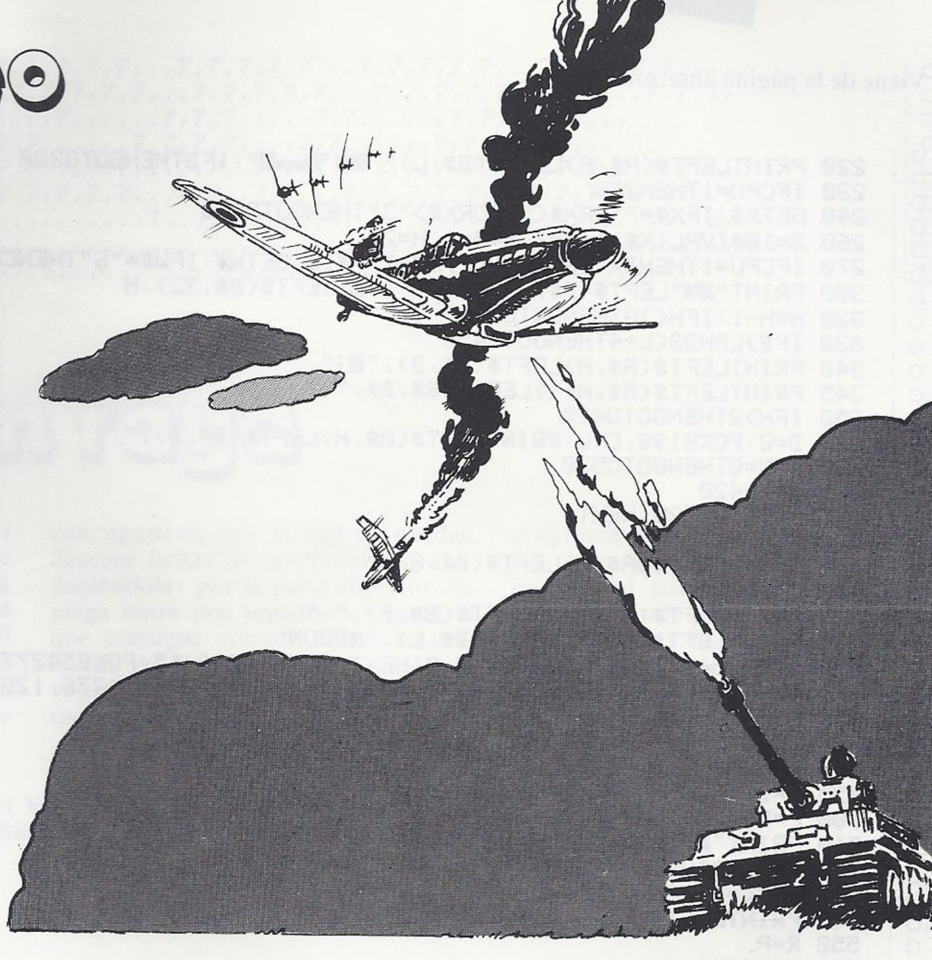
GUIA PRACTICA



Concurso

Antiaéreo

Lorenzo Miguel Soler nos remite desde Madrid este juego, para el 64, llamado ANTIAEREO. Consiste en derribar los aviones enemigos que sobrevuelan el puesto antiaéreo. Para ello se dispone de una batería de tres cañones situada en la parte inferior de la pantalla. Los cañones están numerados con los números 1, 2 y 3 y para dispararlos hay que pulsar las teclas 1, 2 y 3 respectivamente, según el cañón que se desee disparar. Los aviones sobrevuelan el puesto de izquierda a derecha y a diferentes alturas. La puntuación que se obtiene al alcanzar un avión es mayor cuanto más alto estuviera y también es mayor cuanto más a la izquierda de la pantalla se le alcance. El programa comienza con una presentación realizada por un avión enemigo y luego puede comenzarse el juego pulsando la tecla S. Si no se pulsa en enseguida, el programa comienza él solo con una demostración del juego. Es importante que una vez copiado el juego, y antes de escribir *RUN*, se guarde en cassete, ya que si no podría haber problemas, pues el juego inhabilita las teclas *RUN/STOP RESTORE*. Sólo nos queda decir adelante y buena caza.



CBM 64

```
10 REM M.SOLER * ANTIAEREO * C-64
15 R=25:PRINTCHR$(142)"■■"
20 PRINT"■":POKE53281,1:POKE53280,14:POKE774,255:CPU=1:POKE198,0:POKE809,255
30 FORI=1024TO2023:IFI=1979THEN60
40 IFI>1979THEN80
50 GOTO70
60 PRINT"BOUM■■■■■[I]":GOTO80
70 PRINT" ■■■■■■";
80 READP$:P=VAL(P$):POKEI+54272,P:POKEI,160:GETW$:IFW$="S"THENCPU=0:GOTO100
85 NEXT:PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■          PRESIONAR ■S■ PARA EMPEZAR■"
90 FORH=0TO46:POKE53280,H:FORJ=0TO75:NEXT:NEXT
100 FORI=0TO200:NEXT:FORH=1024TO2023:POKEH,32:NEXT
110 RESTORE
130 PRINT"■":A$="■■■■■■■■■■■■■■■■■■■■■■■■■■"
135 B$="■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■"
140 PRINTA$"■■            1            2            3         "
150 PRINTLEFT$(A$,21)TAB(9)"■-■"TAB(19)"■-■"TAB(29)"■-■"
160 PRINT"■■■■■■■PUNTOS: 0"TAB(23)"MUNICION: 30":D=0:P=0:M=30
200 A=INT(RND(0)*14)+2
210 FORL=0TO36
```

Concurso



Viene de la página anterior

```
220 PRINTLEFT$(A$,A)LEFT$(B$,L)" ■■▀▀▀▀" :IFDTHENGOTO320
230 IFCPU=1THEN260
240 GETX$:IFX$=""ORX$<"1"ORX$>"3"THENGOTO400
260 B=10*(VAL(X$)):D=1:M=M-1:H=20
270 IFCPU=1THENB=(INT(RND(1)*3+1))*10:GETW$:IFW$="S"THENCPU=0:GOTO100
300 PRINT"■■"LEFT$(B$,32);"      ";"■"LEFT$(B$,32);M
320 H=H-1:IFH<>ATHENGOTO340
330 IFB>LANDB<L+4THENGOTO450
340 PRINTLEFT$(A$,H)LEFT$(B$,B);"■|"
345 PRINTLEFT$(A$,H+1)LEFT$(B$,B);" "
350 IFH>2THENGOTO420
370 D=0:POKE198,CPU:PRINTLEFT$(A$,H)LEFT$(B$,B);" "
380 IFM=0THENGOTO530
390 GOTO420
400 FORT=0TO29:NEXT
420 NEXT
430 PRINTLEFT$(A$,A)LEFT$(B$,37)"     "
440 GOTO200
450 PRINTLEFT$(A$,H+1)LEFT$(B$,B);" "
460 PRINTLEFT$(A$,A)LEFT$(B$,L);"■BOUM"
470 FORN=54272TO54296:POKEN,0:NEXT:POKE54296,15:POKE54277,254:POKE54278,204
475 POKE54273,17:POKE54272,37:POKE54276,129:POKE54276,128
480 D=0
485 FORI=0TO1000:NEXT
490 PRINTLEFT$(A$,A)LEFT$(B$,L);"      "
495 B=B-4
500 P=P+(16-A)*(4-((B-6)/10+1))
510 PRINT"■■■■■■■■■■■■■■■■|";P
520 IFM<>0THENGOTO200
530 IFP<=RTHENGOTO570
540 PRINTLEFT$(A$,9);"■■■■■■■■ESTABLECIO NUEVO RECORD"
550 R=P
560 GOTO580
570 PRINTLEFT$(A$,9);"■■■■■■■■■■■■■EL RECORD SIGUE"
580 PRINTLEFT$(A$,11)LEFT$(B$,24-(10+LEN(STR$(R))/2));"EN ";R;" PUNTOS"
590 PRINTLEFT$(A$,13);"■■■■■■■DESEA INTENTARLO DE NUEVO ?"
600 TI$="000000"
610 GETY$:IFY$="S"THENCPU=0:GOTO100
620 IFVAL(TI$)=59ORY$="N"THEN20
630 GOTO610
640 END
1000 DATA,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
1010 DATA,,,,,,,,,,,,,,,,7,7,7,7,7,7,,,7,7,,,7,7,,,7,7,7,7,7,7,,,7,7,,,,,,,,
1020 DATA,,,,,,,7,7,7,7,7,7,,,7,7,7,,7,7,,,7,7,7,7,7,7,,,7,7,,,,,,,
1030 DATA,,,,,,,7,7,,,7,7,,,7,7,7,7,7,7,,,,,7,7,,,,,7,7,,,,,,,,
1040 DATA,,,,,,,7,7,,,7,7,,,7,7,7,7,7,7,,,,,7,7,,,,,,7,7,,,,,,,,
1050 DATA,,,,,,,7,7,7,7,7,7,,,7,7,,7,7,7,,,,,7,7,,,,,,7,7,,,,,,,,
1060 DATA,,,,,,,7,7,,,7,7,,,7,7,,,7,7,,,,,7,7,,,,,,7,7,,,,,,,,
1065 DATA,,,,,,,7,7,,,7,7,,,7,7,,,7,7,,,,,7,7,,,,,7,7,,,,,,,,,,,,,,
1070 DATA,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
1080 DATA,,,,,,,,,,,7,7,7,7,7,7,,,7,7,7,7,7,7,,,7,7,7,7,7,7,,,7,7,7,7,7,7,,,7,7,7,7,7,7
1090 DATA7,,,
1100 DATA,,7,7,7,7,7,7,,,7,7,,,,,,,7,7,,,7,,,7,7,,,,,,7,7,7,7,7,7,,,
1110 DATA,,7,7,,,7,7,,,7,7,,,,,,7,7,,,7,,,7,7,,,,,,7,7,,,7,7,,,
```

PROGRAMA ★ commodore Magazine ★ COMPROBADO

```
1120 DATA,,7,7,,,7,7,,,7,7,7,7,7,,,7,7,7,7,7,,,7,7,7,7,7,,,7,7,,,7,7,,,,
1130 DATA,7,7,7,7,7,7,,,7,7,7,7,7,,,7,7,7,7,7,,,7,7,7,7,7,,,7,7,,,7,7,,,
1140 DATA,,7,7,7,7,7,7,,,7,7,,,,,,7,7,7,,,,7,7,,,,,,,,7,7,,,,7,7,,,
1150 DATA,,7,7,,,7,7,,,7,7,,,,,,7,7,7,7,,,,7,7,,,,,,,,7,7,,,,7,7,,,
1160 DATA,,7,7,,,7,7,,,7,7,7,7,7,,,7,7,,7,7,,,7,7,7,7,7,,,7,7,7,7,7,7,,,
1170 DATA,,7,7,,,7,7,,,7,7,7,7,7,,,7,7,,7,7,,,7,7,7,7,7,,,7,7,7,7,7,7,,,
1180 DATA,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,, (69)
1190 DATA,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
2000 DATA,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
```

Cuatro en raya

J. Peña Benet y X. Rodríguez i Sagarra, dos lectores de Commodore Magazine de Tarragona, nos envían a la limón este juego para el C-64 titulado CUATRO EN RAYA. El juego recuerda a ciertos juegos de plástico que pueden verse en muchos sitios y que consisten en un soporte, con agujeros, por el que se pueden deslizar fichas de dos colores, introduciéndolas por la parte superior. Se juega entre dos jugadores y gana el que consigue colocar cuatro de sus fichas en línea, ya sea horizontal, vertical o diagonal. El juego incluye todas las instrucciones como parte del programa y es muy sencillo de manejar. Además incluye una interesante posibilidad para el caso de que se vaya perdiendo y no se esté de humor para llegar al final de la partida y darle la satisfacción de ganar al contrario.

CBM 64

```
0 REM JUEGO DEL CUATRO EN RAYA
1 REM XAVIER RODRIGUEZ SAGARRA Y JOAN PENA BENET  (28-6-84) TARRAGONA.
2 POKE650,128
3 GOSUB700
5 PRINT"[CLR]":M1=0
6 POKE53281,14
7 U=55376:E=1104
10 FORH=0TO7
20 FORT=0TO39:POKEU+T,0:POKEE+T,67:NEXT
30 U=U+3*40:E=E+3*40
40 NEXT
50 U=55296:E=1024
55 L=93
60 FORI=0TO13
70 FORX=0TO24
75 IF(X+1)/3-INT((X+1)/3)=0THENL=91:GOTO80
78 L=93
80 POKEE+40*X,L:POKEU+40*X,0:NEXT
90 U=U+3:E=E+3
100 NEXT
150 V=1065:V1=55337:Q=1
160 POKEV,81:POKEV1,1
200 GETA$:IFA$=""GOTO200
205 POKEV,32
220 IFA$="E"ANDV=1065GOTO200
230 IFA$="D"ANDV=1101GOTO200
240 IFA$="E"THENV=V-3:V1=V1-3
250 IFA$="D"THENV=V+3:V1=V1+3
255 IFA$="F"THENEND
```

PREMIADO CON 5.000 PESETAS

Concurso

Viene de la página anterior

```
257 IFA$="P"GOTO1000
260 POKEV,81:POKEV1,Z
265 IFA$="C"GOTO300
270 GOTO200
300 K=V:K1=V1:M=0:GOTO500
302 Q=-Q
303 IFQ=1THENZ=13:GOTO305
304 Z=6
305 K=K+120:K1=K1+120
308 M=M+1
310 IFPEEK(K)=160THENPOKE(K-120),160:POKE(K1-120),Z:GOTO 450
320 POKEK,160:POKEK1,Z
325 IFM=7GOTO400
330 FORS1=1TO50:NEXT
335 POKEK,32
340 GOTO305
400 POKEK+1,160:POKEK1+1,Z
405 POKEK-40,160:POKEK1-40,Z
410 POKEK-39,160:POKEK1-39,Z
420 GOTO200
450 POKE(K-119),160:POKE(K1-119),Z
455 POKE(K-40-120),160:POKE(K1-40-120),Z
460 POKE(K-40-119),160:POKE(K1-40-119),Z
465 GOTO200
500 U2=V+120
510 IFPEEK(U2)=160GOTO900
520 GOTO302
700 POKE53281,6:PRINT"■":PRINT"■■■■■CUATRO EN RAYA"
705 PRINT:PRINT
710 PRINT"■■ INSTRUCCIONES DEL JUEGO:"
715 PRINT
720 PRINT" ESTE JUEGO CONSISTE EN PONER CUATRO "
730 PRINT" CUADRADOS EN RAYA ,ESTA RAYA      "
740 PRINT" PUEDE SER HORIZONTAL,VERTICAL O"
750 PRINT" DIAGONAL."
752 PRINT" ESTE JUEGO ES PARA 2 JUGADORES"
755 PRINT:PRINT
760 PRINT"■■■■■TECLAS DEL JUEGO"
765 PRINT
770 PRINT"■■ -E ;IZQUIERDA"
780 PRINT" -D ;DERECHA"
790 PRINT" -C ;PARA DEJAR CAER UN CUADRADO"
800 PRINT" -F ,FINAL"
805 PRINT" -P,PARA SI ESTAS A PUNTO DE PERDER Y NO QUIERES PERDER"
850 PRINT:PRINT
860 PRINT"PARA COMENZAR PULSE UNA TECLA...."
865 PRINT"■"
870 GETS$:IFS$=""GOTO870
880 RETURN
900 S=54272:POKES+24,15:POKES+5,100:POKES+1,25:POKES+0,177:POKES+4,17
910 FORP=1TO100:NEXT
920 POKES+4,0:POKES+5,0:POKES+6,0
950 GOTO200
1000 FORNU=1024TO2023
1010 POKENU,32:NEXT
1020 END
```

Concurso

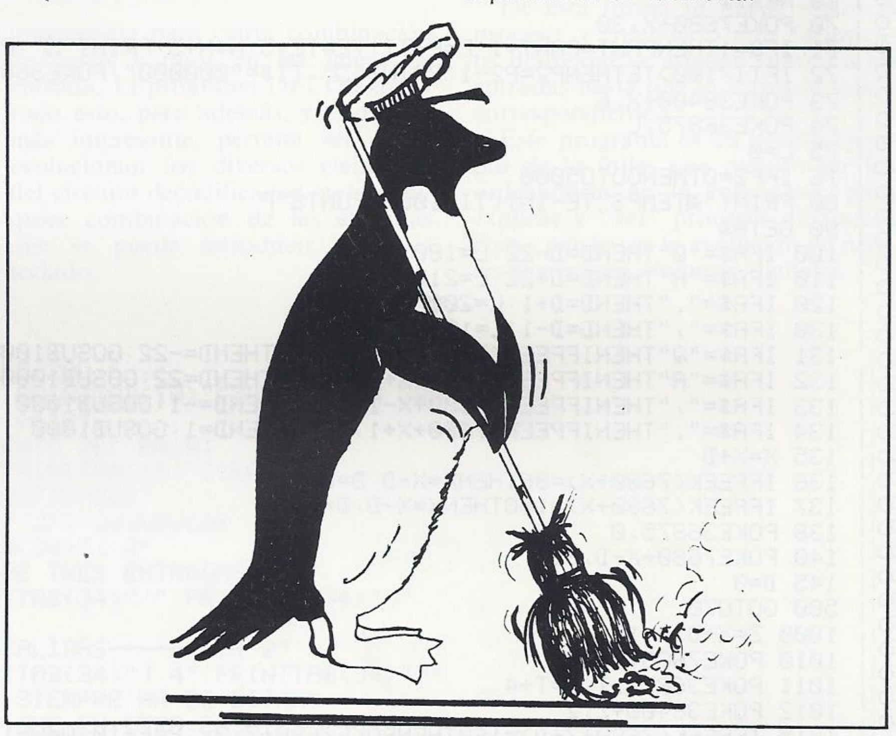
Pingüino

Enric Corberó Serrahina nos envía desde Barcelona este estupendo juego para el **VIC-20**, que él ha titulado PINGUINO. El juego consiste en hacer de empleado del servicio de limpieza y amontonar en un solo bloque toda una serie de paquetes de basura que aparecen aleatoriamente distribuidos por la pantalla. El encargado de amontonar la basura es un pingüino, representado por una flecha y que el jugador controla mediante cuatro teclas. Para mover los bloques basta con acercarse hasta ellos con lo que saldrán disparados, como si les hubiéramos dado una patada bien fuerte. Una de las cosas que hacen muy entretenido a este juego es la velocidad con que hay que mover al pingüino, ya que sólo se dispone de un tiempo limitado para amontonar toda la basura de una pantalla. Además, por cada pantalla se dispone de 3 vidas y si con alguna de estas 3 vidas se consigue amontonar toda la basura, entonces se pasa a una nueva pantalla con más basura que la anterior.

Las teclas con las que se controla al pingüino son:

Q = ARRIBA
A = ABAJO
. = IZQUIERDA
, = DERECHA

El programa es para el **VIC-20** estándar, es decir sin ningún tipo de expansiones de memoria.



VIC 20

```
1 POKE36879,25:PRINT"⌐":POKE650,128:P=0:N=10:PANT=1:P2=3:TE=35:GOSUB10000
2 TI$="000000":POKE36878,15
5 PRINT"■⌐xxxxxxxxxxxxxxxxxxxxxx";
6 FORT=1TO19:PRINT"x                    x";:NEXTT
7 PRINT"xxxxxxxxxxxxxxxxxxxxxx";
8 POKE36876,0
9 Q=N-1
20 X=144:O=130
25 PRINT"■PANTALLA"PANT"TANCS"P2;"⌐⌐"
30 FORA=1TON
35 O=O+3
36 POKE36875,0
40 J=INT(RND(1)*368)+69
44 IFPEEK(7680+J+1)=86ORPEEK(7680+J-1)=86THENGOTO40
45 IFPEEK(7680+J)=86THENGOTO40
50 POKE7680+J,160
55 POKE38400+J,2
60 NEXTA
61 POKE36875,0
62 FORT=1TO8
63 Z=INT(RND(1)*368)+69
64 IFPEEK(7680+Z)<>32THEN63
65 POKE7680+Z,86
```

Viene de la página anterior

```
66 NEXTT
70 POKE7680+X,30
71 IFQ=1THENPANT=PANT+1:P=P+TI:TE=TE+5:N=N+5:PRINT"□":POKE36876,128:GOTO2
72 IFTI/100>TETHENP2=P2-1:PRINT"□":TI$="000000":POKE36876,211:GOTO5
73 POKE38400+X,6
74 POKE36875,L
75 L=0
76 IFP2=0THENGOTO5000
80 PRINT"■TEMPS"TE-INT(TI/100)"PUNTS"P
90 GETA$
100 IFA$="Q"THEND=D-22:L=180
110 IFA$="A"THEND=D+22:L=210
120 IFA$="."THEND=D+1:L=200
130 IFA$=","THEND=D-1:L=190
131 IFA$="Q"THENIFPEEK(7680+X-22)=160THEND=-22:GOSUB1000
132 IFA$="A"THENIFPEEK(7680+X+22)=160THEND=22:GOSUB1000
133 IFA$=","THENIFPEEK(7680+X-1)=160THEND=-1:GOSUB1000
134 IFA$="."THENIFPEEK(7680+X+1)=160THEND=1:GOSUB1000
135 X=X+D
136 IFPEEK(7680+X)=86THENX=X-D:D=0
137 IFPEEK(7680+X)=160THENX=X-D:D=0
138 POKE36875,0
140 POKE7680+X-D,32
145 D=0
500 GOTO70
1000 Z=X+D:T=170
1010 POKE7680+Z,160
1011 POKE36877,T:T=T+4
1012 POKE38400+Z,2
1015 IFPEEK(7680+Z+D)=160THENPOKE7680+Z,32:P=P+10:Q=Q-1:POKE36877,0:RETURN
1020 POKE7680+Z,32
1026 IFPEEK(7680+Z+D)=86THENPOKE7680+Z-D,160:POKE36877,0:RETURN
1030 Z=Z+D
1050 GOTO1010
1500 GOTO70
5000 PRINT"■PUNTS"P"MANQ."Q:POKE36878,0
5001 PRINT"■■■■■■■■■■■■GAME OVER■"
5010 IFP>RETHENRE=P
5020 PRINT"■■■■■■■■RECORD:"RE
5023 FORA=1TO10:GETA$:NEXT
5024 IFA$<>""THEN1
5025 GOTO5023
10000 PRINT"■■■■■■ENRIC CORBERO PRODUCTIONS PRESENTS"
10001 PRINT"■■■■■■■■■PINGUINO"
10010 PRINT"■■■■■■DEBERAS RECOJER TODA LA BASURA QUE ENCUENTRES, PARA ELLO CONTR
OLARAS "
10020 PRINT"A TU PINGUINO CON:"
10030 PRINT"■■■■■■■Q: ARRIBA"
10031 PRINT"■■■A: ABAJO"
10032 PRINT"■■■K: IZQUIERDA"
10033 PRINT"■■■D: DERECHA"
10050 FORT=1TO4000:NEXTT:RETURN
```

PROGRAMA commodore COMPROBADO

PREMIADO CON 5.000 PESETAS



DECO

CBM - 64

DECO es un programa didáctico para el **C-64** que nos remite José Manuel García Pérez desde Madrid. El programa explica de una forma clara y amena el funcionamiento de un circuito decodificador. Un decodificador es un circuito electrónico con varias líneas de salida, cada una de las cuales se activa (cambia de 0 a 1 ó viceversa según el tipo de que se trate) únicamente para cierta combinación de ceros y unos en las líneas de entrada. El programa DECO explica todo esto, pero además, y esto es lo más interesante, permite ver como evolucionan los diversos elementos del circuito decodificador, para cualquier combinación de las entradas que se puede introducir desde el teclado.

De esta forma se puede seguir el proceso, y entenderlo muy fácilmente, desde que se asignan valores a las entradas hasta que se activa la salida correspondiente.

Este programa es un bonito ejemplo de lo útiles que pueden ser los ordenadores en la enseñanza, para explicar y "ver" procesos dinámicos como puede ser la evolución de niveles lógicos en circuitos digitales.

```
10 PRINT"♥";:PRINT"♥";:POKE53280,6
20 PRINT:PRINT:PRINT:PRINT:PRINTTAB(10)"EXPLICACION GRAFICA"
30 PRINT:PRINTTAB(18)"DEL":PRINT
40 PRINTTAB(11)"FUNCIONAMIENTO DE":PRINT
50 PRINTTAB(18)"UN":PRINT:PRINTTAB(15)"CIRCUITO"
60 PRINT:PRINTTAB(13)"DECODIFICADOR"
70 FORT=1TO7000:NEXTT:PRINT"♥";:GOSUB9600
80 PRINTTAB(34)" ╱":PRINTTAB(34)"|.A"
90 PRINT"  EL CIRCUIT@ TIENE TRES ENTRADAS ┤ B"
100 PRINTTAB(34)"| C":PRINTTAB(34)" ╲":PRINTTAB(34)" ╱"
110 PRINTTAB(34)"| 1"
120 PRINTTAB(10)"Y CUATRO SALIDAS─────────┤ 2"
130 PRINTTAB(34)"| 3":PRINTTAB(34)"| 4":PRINTTAB(34)" ╲"
140 PRINT" -LA ENTRADA (A) SIEMPRE HA DE ESTAR"
150 PRINT"  EN ESTADO 1 YA QUE EN OTRO CASO"
160 PRINT"  LA SALIDA POR 1,2,3 O 4 SERA CERO"
180 PRINT" -LAS SALIDAS SERAN:":PRINTTAB(21)"B C     S"
190 PRINTTAB(21)"- -     -":PRINTTAB(21)"0 0────1"
200 PRINTTAB(21)"0 1────2":PRINTTAB(21)"1 0────3"
210 PRINTTAB(21)"1 1────4":PRINT
220 PRINT"  (PARA CONTINUAR PULSE UNA LETRA)"
230 GETA$:IFA$=""THEN230
233 PRINT"♥";:PRINT"INTRODUZCA LOS VALORES DE B Y C"
235 INPUT"DE LA FORMA:B,C(SOLO CERO O UNO)";B,C
240 PRINT"♥";:GOSUB2990
250 POKE1306,1:POKE1666,2:POKE1826,3:POKE1139,49
260 POKE1339,50:POKE1539,51:POKE1739,52
270 PRINT"B=";B;"C=";C:FORY=1TO1000:NEXTY
290 B=B+1:ONBGOSUB5000,7000
300 PRINT"PARA CONTINUAR PULSE UNA LETRA"
310 POKEX,2:FORT=1TO200:NEXTT:POKEX,1:FORT=1TO200:NEXTT
320 GETB$:IFB$=""THEN310
340 PRINT"SI QUIERE OTRO EJEMPLO PULSE LA LETRA P"
350 INPUT"SI NO PULSE OTRA CUALQUIERA";N$
360 PRINT"♥";:IFN$="P"THEN233
370 FORE=1TO9:PRINT:NEXTE:PRINTTAB(17)"FIN"
380 END
2990 POKE53281,14
3000 PRINT
```

Concurso



Viene de la página anterior

```
3010 PRINT"          ┌──┐            ┌──┐"
3020 PRINT"       ┌──┤* ├─────┬──────┤* ├───●"
3030 PRINT"       │ ┌┤  │     │ ┌────┤  │"
3040 PRINT"       │ │└──┘     │ │    └──┘"
3050 PRINT"       │ │         │ │"
3060 PRINT"       │ │         │ │    ┌──┐"
3070 PRINT"   ────┤ │         └─┼────┤* ├───●"
3080 PRINT"       │ │           │  ┌─┤  │"
3090 PRINT"       │ │           │  │ └──┘"
3100 PRINT"       │ │           │  │"
3110 PRINT"       │ │ ┌──┐    │ │  │ ┌──┐"
3120 PRINT"       └─┼─┤* ├────┼─┼──┼─┤* ├───●"
3130 PRINT"         │┌┤  │    │ ├──┼─┤  │"
3140 PRINT"      ┌─┐││└──┘    │ │  │ └──┘"
3150 PRINT"     ┌┤-├┘│        │ │  │"
3160 PRINT"   ──┤└─┘ │        │ │  │"
3170 PRINT"     └────┘        └─┼──┼─┤* ├───●"
3180 PRINT"      ┌─┐            │  ├─┤  │"
3190 PRINT"     ┌┤-├────────────┘  │ └──┘"
3200 PRINT"   ──┤└─┘              │"
3210 PRINT"     └──────────────────┘"
3220 FORY=1TO400:NEXTY:POKE53281,6:RETURN
5000 IFC=0THENX=55409
5010 IFC=1THENX=55609:GOTO6000
5030 GOSUB9000
5040 GOSUB9100
5050 GOSUB9200
5060 FORY=1TO10:POKE55404,2:FORZ=1TO200:NEXTZ
5070 POKE55404,14:FORZ=1TO200:NEXTZ:NEXTY
5080 FORY=55406TO55408:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY:GOTO6900
6000 GOSUB9000
6010 GOSUB9100
6020 GOSUB9400
6030 FORY=1TO10:POKE55604,2:FORZ=1TO200:NEXTZ
6040 POKE55604,14:FORZ=1TO200:NEXTZ:NEXTY
6050 FORY=55606TO55608:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
6900 RETURN
7000 IFC=0THENX=55809
7010 IFC=1THENX=56009:GOTO8000
7020 GOSUB9000
7030 GOSUB9500
7040 GOSUB9200
7050 FORY=1TO10:POKE55804,2:FORZ=1TO200:NEXTZ:POKE55804,14:FORZ=1TO200
7055 NEXTZ:NEXTY
7060 FORY=55806TO55808:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY:GOTO8900
8000 GOSUB9000
8010 GOSUB9500
8020 GOSUB9400
8030 FORY=1TO10:POKE56004,2:FORZ=1TO200:NEXTZ:POKE56004,14:FORZ=1TO200:NEXTZ
8035 NEXTY
8040 FORY=56006TO56008:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
8900 RETURN
9000 POKE55579,2:FORZ=1TO200:NEXTZ:POKE55580,2:FORY=1TO150:NEXTY
9010 POKE55581,2:FORY=1TO150:NEXTY:POKE55582,2:FORY=1TO150:NEXTY
9020 POKE55542,2:POKE55622,2:FORY=1TO148:NEXTY:POKE55502,2:POKE55662,2
```



```
9030 FORY=1TO148:NEXTY:POKE55702,2:POKE55462,2:FORY=1TO148:NEXTY
9040 POKE55422,2:POKE55742,2:FORY=1TO148:NEXTY:POKE55782,2:POKE55382,2
9050 FORY=1TO148:NEXTY:POKE55383,2:POKE55783,2:FORY=1TO148:NEXTY
9060 POKE55384,2:POKE55784,2:FORY=1TO150:NEXTY:POKE55785,2
9070 RETURN
9100 FORY=1TO10:POKE55902,2:FORZ=1TO200:NEXTZ:POKE55902,14:FORZ=1TO200
9105 NEXTZ:NEXTY
9110 POKE55904,2:FORY=1TO13:POKE55904-40*Y,2:FORZ=1TO150:NEXTZ:NEXTY
9120 FORY=1TO10:POKE55387,2:FORZ=1TO200:NEXTZ
9130 POKE55387,14:FORZ=1TO200:NEXTZ:NEXTY
9140 FORY=55389TO55401:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9150 FORY=55435TO55595STEP40:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9160 FORY=55596TO55601:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9190 RETURN
9200 FORY=1TO10:POKE56062,2:FORZ=1TO200:NEXTZ
9210 POKE56062,14:FORZ=1TO200:NEXTZ:NEXTY
9220 FORY=56064TO56077:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9230 POKE56037,2:FORZ=1TO300:NEXTZ:POKE55957,2:FORZ=1TO150:NEXTZ
9240 POKE55917,2:FORZ=1TO150:NEXTZ:POKE55877,2:FORZ=1TO150:NEXTZ
9245 POKE55837,2:FORZ=1TO150:NEXTZ:POKE55838,2:FORZ=1TO150:NEXTZ
9250 POKE55757,2:POKE55839,2:FORZ=1TO148:NEXTZ:POKE55717,2
9260 FORY=1TO150:NEXTY:POKE55677,2:POKE55841,2:FORY=1TO150:NEXTY
9270 POKE55637,2:FORY=1TO150:NEXTY:POKE55557,2:FORY=1TO150:NEXTY
9280 POKE55517,2:FORY=1TO150:NEXTY:POKE55477,2:FORY=1TO150:NEXTY
9290 FORY=55437TO55441:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9295 RETURN
9400 POKE56099,2:FORY=1TO150:NEXTY:POKE56100,2:FORY=1TO150:NEXTY
9410 POKE56060,2:POKE56140,2:FORY=1TO150:NEXTY
9420 FORY=56140TO56160:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9430 FORY=56120TO56040STEP-40:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY:POKE56041,2
9450 FORZ=1TO150:NEXTZ
9460 FORY=55960TO55640STEP-40:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9470 POKE55641,2:RETURN
9500 POKE55939,2:FORY=1TO150:NEXTY:POKE55940,2:FORY=1TO150:NEXTY
9510 POKE55900,2:POKE55980,2:FORY=1TO148:NEXTY
9520 FORY=55981TO55984:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9530 FORY=55985TO55825STEP-40:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9540 FORY=1TO10:POKE55788,2:FORZ=1TO200:NEXTZ:POKE55788,14:
9545 FORZ=1TO200:NEXTZ:NEXTY
9550 FORY=55790TO55799:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY:FORY=1TO300:NEXTY
9560 POKE55801,2:FORY=55835TO55995STEP40:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9570 FORY=55996TO56001:POKEY,2:FORZ=1TO150:NEXTZ:NEXTY
9580 RETURN
9600 PRINTTAB(5)"UN CIRCUITO DECODIFICADOR USA"
9610 PRINTTAB(5)"DOS TIPOS DE PUERTAS LOGICAS"
9620 PRINT:PRINT:PRINTTAB(3)"-PUERTA Y (*)="
9630 PRINTTAB(5)"EN LA QUE SOLO HAY SALIDA SI TODAS"
9640 PRINTTAB(5)"LAS ENTRADAS ESTAN EN NIVEL 1"
9650 PRINT:PRINTTAB(3)"-INVERSOR (-)="
9660 PRINTTAB(5)"EL INVERSOR O COMPLEMENTO ES UN"
9670 PRINTTAB(5)"CIRCUITO CUYA SALIDA ES LA INVERSA"
9680 PRINTTAB(5)"DE SU ENTRADA."
9690 FORY=1TO9:PRINT:NEXTY:PRINT"(PARA CONTINUAR PULSAR UNA LETRA)"
9700 GETA$:IFA$=""THEN9700
9710 A$="":PRINT"";:RETURN
```

Concurso

En busca del tesoro

Ignacio Parellada y **Ricardo Portabelda** nos envían desde Barcelona el programa **en busca del tesoro,** para el *Vic 20* en su versión estándar. El juego consiste en desplazarse a través de un laberinto aleatorio hasta llegar a donde se encuentra el tesoro, representado por el símbolo $, y hacerse con él. Hay, sin embargo, dos obstáculosd que salvar antes de poder llegar al tesoro; el primero de ellos es el tiempo, ya que sólo hay un minuto para conseguir el dinero en cada pantalla. Pero además, puede ocurrir que el tesoro esté al otro lado de una pared, en cuyo caso hay que hacer uso de un pico para romper dicha pared, teniendo en cuenta que sólo puede hacerse un agujero por cada laberinto. Cada vez que se consigue el tesoro aparecerá una nueva pantalla, con un nuevo tesoro que recoger, pero también con mayor número de obstáculos que superar. El programa incluye instrucciones sobre las teclas que hay que utilizar para moverse por el laberinto.



VIC 20

```
1 POKE650,255:GOSUB10000
2 PRINT"■":POKE36879,12:II=0:OO=0:GOSUB7000
6 PRINT"□■";:V=1:X=0:F=0:D=0:OO=OO+.1
7 W=0:X= RND(1)*10:Z=8174:AA=0
8 IFX<=OOTHEN10
9 IFX>OOTHEN20
10 PRINTCHR$(166);:F=F+1:GOTO35
20 PRINTCHR$(160);
30 D=D+1:GOTO35
35 IFD+F<487THEN7
39 FORTT=1TO4
40 Y=INT(RND(1)*466)+1:N=7680
60 IFPEEK(N+Y+U)=102ORPEEK(N+Y+U)=30ORPEEK(Y+U+N)=36THENU=U+1:GOTO40
65 POKE7680+Y+U,36
70 NEXT
80 POKE8174,81
85 POKE8152,32:POKE8130,32
87 Z=8174
88 TI$="000000"
89 POKEZ+6,30
90 GETA$:IFA$=""THEN90
```

PREMIADO CON 5.000 PESETAS

```
91 IFVAL(TI$)>100THENGOTO6000
95 POKEZ+AA,32
100 IFA$="I"THENAA=AA-22:GOTO135
110 IFA$="M"THENAA=AA+22:GOTO135
120 IFA$="L"THENAA=AA+1:GOTO135
130 IFA$="J"THENAA=AA-1:GOTO135
135 IFPEEK(Z+AA)=102ANDV=1THENGOSUB1000
136 IFPEEK(Z+AA)=102ANDV=0THENV=1:POKEZ+6,30
137 IFPEEK(Z+AA)=30THENGOSUB2000
138 IFPEEK(Z+AA)=36THENGOSUB3000
139 IFZ+AA>8186THENAA=AA-22:GOSUB500
145 IFZ+AA<7680THENAA=AA+22:GOSUB500
190 POKEZ+AA,81
200 GOTO90
500 POKE36878,15
510 FORX=1TO25
520 POKE36876,200
530 NEXT
540 POKE36876,0:RETURN
1000 POKE36878,15
1010 FORX=1TO200
1020 POKE36876,200
1030 NEXT
1035 POKE36876,0
1040 POKEZ+AA,102:AA=0:RETURN
2000 POKE36878,15
2010 FORL=1TO10
2015 FORM=250TO240STEP-1
2020 POKE36876,M
2030 NEXTM
2040 FORM=240TO250
2050 POKE36876,M
2060 NEXTM
2070 POKE36876,0
2080 NEXTL
2090 POKE36878,0
2100 V=0:RETURN
3000 W=W+1:SC=SC+5*(II+1)*W+(100-VAL(TI$))*TT
3010 IFW=4THENPOKEZ+AA,81:PRINT"■QQQ"SC" PUNTOS":FORBB=1TO4000:NEXT:GOSUB8000:D
=0:GOTO6
3020 POKE36878,15
3030 FORI=128TO255
3040 POKE36875,I
3050 FORK=1TO3:NEXT
3060 NEXT
3070 POKE36875,0
3080 RETURN
6000 PRINT"■QQQQQQQQQ▶SE TERMINO EL TIEMPO"
6005 FORYY=1TO500:NEXT
6010 R1=36875:R2=36878:POKER2,15
6020 READPA:IFPA=-1THEN6100
6030 READDF:POKER1,PA:FORNM=1TODF:NEXT
6040 POKER1,0:FORMN=1TO20:NEXT:GOTO6020
6100 POKER2,0
6200 DATA232,400,225,150,225,150,228,250,225,750,231,300,232,300,-1
6500 PRINT" QQQ▶▶▶▶"SC" PUNTOS":FOREE=1TO5000:NEXT
6620 SC=0:RESTORE:PRINT"■":GOTO2
7000 PRINT"■QQPARA TERMINAR /0/ ":PRINT:INPUT"NIVEL (1-3)";TT
7005 IFTT=0THENPRINT"■":END
```

Concurso

Viene de la página anterior

```
7010 IFTT>3ORTT<0THEN7000
7020 OO=TT+2.5
7025 GOSUB8000
7030 RETURN
8000 II=II+1:PRINT"■■■■■■■■■■■■■■■LABERINTO "II:FORRR=1TO5000:NEXT
8010 PRINT"■":RETURN
10000 POKE36879,46
10010 PRINT"■■"
10020 PRINT"■■■EN BUSCA DEL TESORO!■"
10030 PRINT"■■MOVIMIENTO:/I/J/L/M/"
10040 PRINT"■TIEMPO:1MINUTO"
10050 PRINT"■OBJETIVO:RECOGER ($)
10060 PRINT"■DISPONES DE UN PICO
10070 PRINT"■PARA ABRIR UNA BRECHA.
10080 PRINT"A MAYOR CANTIDAD DE
10090 PRINT"■LABERINTOS MAYOR ES
10100 PRINT"■LA DIFICULTAD.
10105 PRINT"■      PULSA /F1/
10110 GETS$:IFS$<>CHR$(133)THEN10110
10120 RETURN
```

Concurso

Histogramas

Francisco Guindos Rojas, lector de **Commodore Magazine** de **Granada**, nos manda esta pequeña pero interesante rutina para confeccionar histogramas con el CBM 64. Los histogramas o diagramas de barras consisten en la representación de un conjunto de valores mediante un gráfico de barras de diferentes alturas. Cada barra representa un valor diferente, y su altura es mayor cuanto mayor sea el valor que representa.

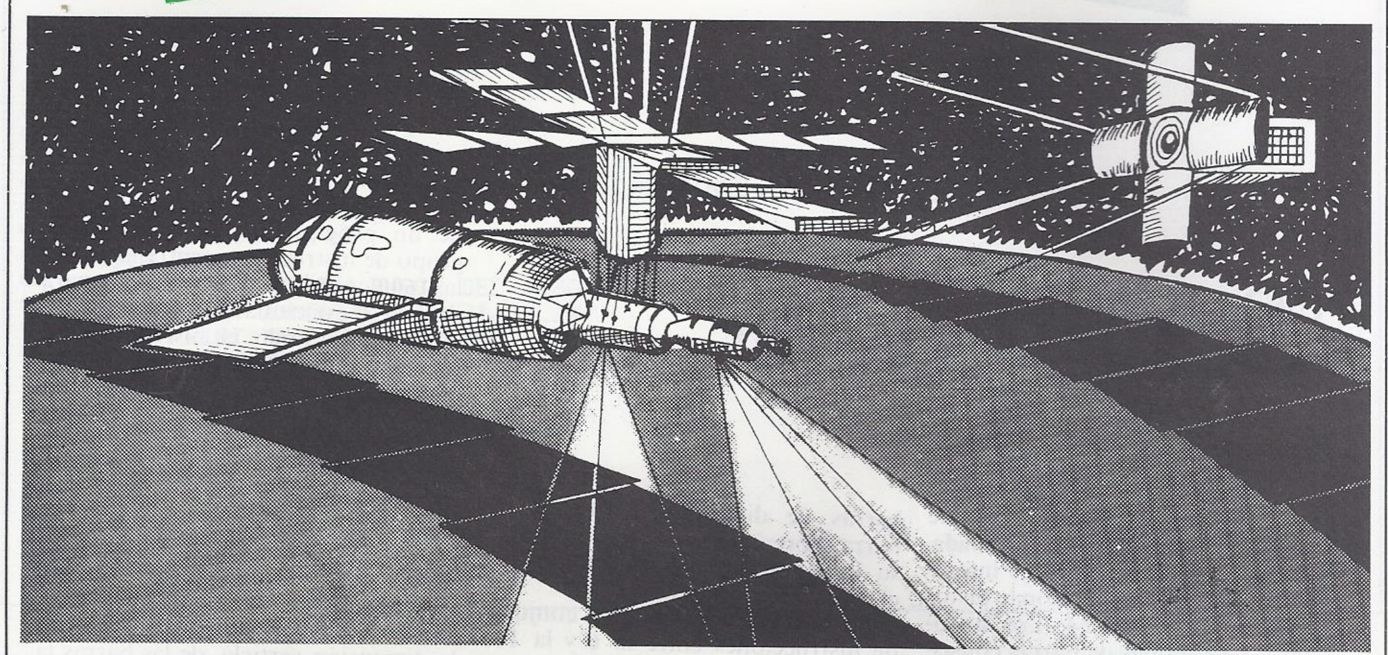
La rutina comprende el conjunto de instrucciones entre la **6** y la **160**, ambas inclusive, y está pensada para ser utilizada como subrutina dentro de un programa más complejo. El grupo de instrucciones entre la **1000** y la **1600** constituyen un ejemplo para ilustrar el funcionamiento de la rutina, representando el histograma de un conjunto aleatorio de valores, tantos como se desee hasta un máximo de **38**. Para utilizar la rutina dentro de otro programa hay que tener en cuenta que, antes de llamarla con un GOSU, es necesario definir el número de datos a representar y asignarlo a la variable **N** y también es necesario que los datos vayan almacenados en una matriz **A(I)** de dimensión N. El programa aprovecha, para la dimensión verticla, de las barras la máxima resolución gráfica del **64**.




```
0 REM          HISTOGRAMAS
1 REM
2 REM      AUTOR   F. GUINDOS
3 REM
5 GOTO1000
6 DATA164,175,185,162,184,183,163
10 POKE53280,0:AM=INT(38/N):A$="":FORF=1TO7:READG:A$=A$+CHR$(G):NEXT
20 M=0:FORF=1TON:IFM<A(F)THENM=A(F)
30 NEXT
40 FORF=1TON:A(F)=A(F)*186/M:NEXT
50 PRINT"]"
60 P=INT((41-N*AM)/2)+1
70 FORF=1TON:FORG=1TOAM
80 X=P+(F-1)*AM+G-1:Y=23:IFX=0THENX=256
90 POKE211,X-1:POKE214,Y:SYS58732
100 IFA(F)<8THEN120
110 FORH=1TOINT(A(F)/8):PRINTCHR$(18);" ";CHR$(145);CHR$(157);CHR$(146);:NEXT
120 IF(INT(A(F))AND7)>4THENPRINTCHR$(18);
130 IF(INT(A(F))AND7)=0THEN150
140 PRINTMID$(A$,(A(F)AND7),1)
150 NEXT:NEXT
160 RETURN
1000 PRINTCHR$(147)TAB(93)"HISTOGRAMAS":PRINT:PRINT
1100 INPUT"NUMERO DE DATOS (MAX=38)";N
1200 IFN<>INT(ABS(N))ORN>38THEN1000
1300 DIMA(N)
1400 FORF=1TON:A(F)=(.5+RND(1))*F*(N-F+1):NEXTF
1500 GOSUB10
1600 GETA$:IFA$=""THEN1600
```

PROGRAMA COMPROBADO commodore Magazine

Concurso



Satélites artificiales

CBM 64

Francesc Pinyol, al parecer muy aficionado a la mecánica celeste, concursa con este programa llamado Satélites Artificiales. Según nos aclara, el programa consta de dos partes. En la primera se introducen los datos que determinan la trayectoria del satélite, calculándose la excentricidad de la órbita. Después podemos indicar si deseamos ver representada la órbita seguida. Contestando afirmativamente se pasa a la segunda parte del programa.

Para representar la trayectoria hay que contestar cinco preguntas que aparecen en la pantalla. En primer lugar hay que borrar la órbita anterior, si es la primera vez que se quiere ver desde que se ha cargado el programa. En las siguientes no es necesario, pudiéndose comparar las diferentes órbitas anteriores.

La escala se elige entre dos opciones. En la primera cada *pixel* representa **530 kilómetros** y en la segunda es la mitad. La última pregunta que hace el ordenador es curiosa. Podemos ver las estrellas como parte del fondo.

Lo cierto es que cuando se corre por vez primera el programa, el despiste en cuanto a las magnitudes factibles es total. El autor lo acompaña con varios ejemplos para que no surja la frustración incial. Veamos alguno: Altura en Km.: 10; Vel. inic. en m/s.: 11.000; g del planeta: 9.8; radio: 6.370. Segundo ejemplo: Altura en Km.: 15.000; vel. inic. en m/s.: 5.000; g del planeta: 9.8; radio del planeta: 6.370, etc.

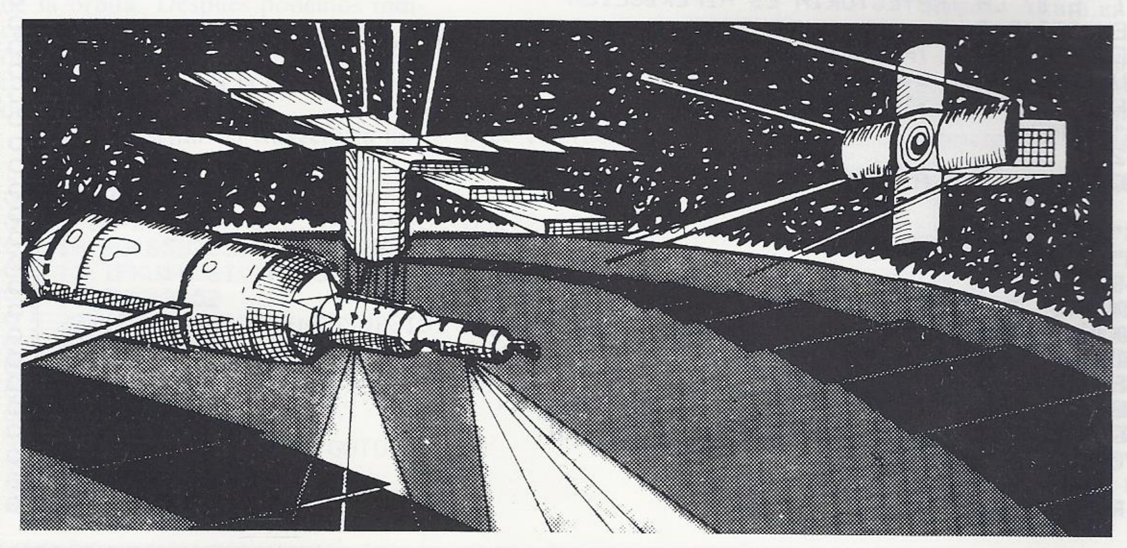
```
2 GOSUB7000:GOSUB2500
5 BA=2*4096:POKE53272,PEEK(53272)OR8
10 POKE53265,PEEK(53265)OR32
13 IFKU$="SI"THENSYS49178
30 SYS49152
33 IFES$="SI"THENGOSUB8191
34 POKE53280,12:QQ=0:GOSUB8000
35 IFPU=2THEN6004
50 FORX=320TO0STEP-2:Y2=WO*((1-A*(X-200))↑2)-(X-200)↑2:IFY2<0THENNEXT
65 XU=X
66 IFSE=1THENW=-1:GOTO148
67 IFSE=2THENW=1
148 SA=0:M=-1:FO=XU:GOTO150
```

```
149 SA=XU:M=1:FO=XO+1
150 FORX=FOTOSASTEPM*T:Y2=WO*((1-A*(X-200))↑2)-(X-200)↑2:IFY2<0THEN197
190 Y=(SQR(Y2)*M)+100:IFY>200ORY<0THENNEXT:GOTO193
192 G=BA+INT(Y/8)*320+8*INT(X/8)+(YAND7):Z=PEEK(G)OR(2↑(7-XAND7)):POKEG,Z:NEXT
193 IFM=-1THENY=200-Y:GOTO197
195 POKE1024,3:GOSUB10000
196 GOTO196
197 W=-W:XO=X:GOTO149
2500 PRINT"████████████████SATELITES ARTIFICIALES█"
2501 PRINT"████████████████F.PINYOL":PRINT"████████████████SOFTWARE"
2502 QQ=1:GOSUB8000:POKEV+2,250
2503 PRINT"████████████****PULSA UNA TECLA PARA CONTINUAR*****█"
2504 S=54272:FORL=STOS+24:POKEL,0:NEXT:POKES+5,15:POKES+1,6:POKES,16:POKES+4,129
2506 FORL=255TO0STEP-1:POKEV+3,L:POKES+24,(L/17):GETX$:IFX$<>""THEN2513
2512 NEXT:POKES+4,0:GOTO2504
2513 POKES+4,0:POKEV+2,0:POKEV+3,0
3000 PRINT"██INTRODUCE LOS DATOS SIGUIENTES:"
3001 INPUT"███ LA ALTURA EN KM";H:H=H/530
3002 INPUT"█ V0 TANGENCIAL EN M/SEG";V0:V0=V0/530000
3003 INPUT"█ G DEL PLANETA (TIERRA=9.8)";G:IFG=0THENG=9.8
3004 INPUT"█ RADIO PLANETA EN KM (T=6370)";RA:IFRA=0THENRA=6370
4000 G=G/530000:RA=RA/530
4005 R0=H+RA
4010 MU=G*((RA)↑2)
4015 K=R0*V0
4016 WO=((K↑2)/MU)↑2
4020 A=(R0*(V0↑2)-MU)/((R0↑2)*(V0↑2))
4025 E=((R0*(V0↑2))-MU)/MU
4026 PRINT"██LA EXCENTRICIDAD (E) ES";E
4027 FORF=0TO600:NEXT
4028 IFE>1ORE=1THENPRINT"██**EL SATELITE SE PIERDE EN EL ESPACIO**█"
4029 IFE<0THENPRINT"██**EL SATELITE CAE SOBRE EL PLANETA**█"
4030 IFE<1ANDE>0THENPRINT"██**EL SATELITE ENTRA EN ORBITA ELIPTICA**█"
4031 IFE=0THENPRINT"██**EL SATELITE ENTRA EN ORBITA CIRCULAR**█"
4032 IFE=1THENPRINT"██*EL SATELITE ENTRA EN ORBITA PARABOLICA*█"
4034 PRINT"█SI  E<0  NO HAY TRAYECTORIA,( CAE )"
4035 PRINT"SI  E=0  LA TRAYECTORIA ES CIRCULAR"
4036 PRINT"SI 0<E<1 LA TRAYECTORIA ES ELIPTICA"
4037 PRINT"SI  E=1  LA TRAYECTORIA ES PARABOLICA"
4038 PRINT"SI  E>1  LA TRAYECTORIA ES HIPERBOLICA"
4039 IFE>0THENGOTO4047
4040 IFE<0THENPRINT"██LA TRAYECTORIA NO SE PUEDE REPRESENTAR*"
4041 PRINT"QUIERES HACER OTRO LANZAMIENTO ?"
4042 GETX$:IFX$="S"THENRESTORE:GOTO3000
4043 IFX$="N"THENEND
4044 GOTO4042
4047 PRINT"█QUIERES VER LA TRAYECTORIA DEL SATELITE?"
4048 GETG$
4049 IFG$="S"THEN5000
4050 IFG$="N"THEN4041
4051 GOTO4048
5000 PRINT"███DATOS PARA LA REPRESENTACION GRAFICA:"
5001 INPUT"███QUIERES BORRAR LAS ANTERIORES ORBIT";KU$
5015 INPUT"█DENSIDAD DE PUNTOS (DE 1.5 A 5)";T
5018 PRINT"█ESCALA: 1 (1PIXEL=530KM)":INPUT"         2:(1PIXEL=265KM)";PU
5020 INPUT"█SENTIDO DE LA TRAYECTORIA (1 o 2)";SE
5021 INPUT"█QUIERES ESTRELLAS DE FONDO";ES$:RETURN
6004 H=H*2:V0=V0*2:G=G*2:RA=RA*2
6005 R0=H+RA
6010 MU=G*((RA)↑2)
```

Concurso

```
6015 K=R0*V0
6016 WO=((K↑2)/MU)↑2
6020 A=(R0*(V0↑2)-MU)/((R0↑2)*(V0↑2))
6025 GOTO50
7000 A=49152:FORN=0TO25:READD%:POKEA+N,D%:NEXT
7001 A=49178:FORN=0TO25:READD%:POKEA+N,D%:NEXT:RETURN
7050 DATA169,16,162,0,134,150,162,4,134,151,160,0,145,150
7055 DATA200,208,251,230,151,166,151,224,8,208,243,96
7060 DATA169,0,162,0,134,150,162,32,134,151,160,0,145,150
7065 DATA200,208,251,230,151,166,151,224,64,208,243,96
8000 V=53248:POKEV+21,6:POKE2041,14:IFQQ=0THEN8005
8001 FORN=0TO62:READQ:POKE896+N,Q:NEXT
8005 POKEV+40,1:IFQQ=1THEN9040
8010 POKEV+16,2:POKEV+2,40:POKEV+3,60
8020 DATA0,56,0,0,68,0,0,84,0,0,68,0,0,68,0,0,68,0
8025 DATA0,198,0,0,198,0,0,198,0,0,198,0
8030 DATA1,199,0,1,69,0,2,68,128,12,68,96,16,84,16
8035 DATA96,124,12,128,146,2,153,17,50,129,17,2,255,57,254,1,215,0
8040 IFQQ=1THENRETURN
8041 GOTO9000
8191 FORX=0TO320:Y=RND(1)*200:CH=INT(X/8):RO=INT(Y/8):LN=YAND7
8192 BY=BA+RO*320+8*CH+LN:BI=7-(XAND7):POKEBY,PEEK(BY)OR(2↑BI):NEXT
8193 RETURN
9000 V=53248:POKEV+21,6:POKE2042,15
9001 FORN=0TO62:READQ:POKE960+N,Q:NEXT
9006 IFPU=1THENPOKEV+4,214:POKEV+5,140
9007 IFPU=2THENPOKEV+4,200:POKEV+5,130:POKEV+23,4:POKEV+29,4
9010 POKEV+41,1:POKEV+28,4:POKEV+37,14:POKEV+38,6
9020 DATA0,100,0,2,86,0,9,114,128,6,101,64,25,119,144,46,150,112,119,101,76
9025 DATA89,123,232,150,14,180,180 ,186,120,86,86,220,231,213,252,100,126,60
9030 DATA127,29,248,238,251,212,51,127,240,62,126,224,7,185,192,1,213,64,,121
9040 DATA,,,,
9050 RETURN
10000 S=54272:POKES+5,65:POKES+1,43:POKES,52:POKES+24,15:POKES+4,17:POKES+6,120
12503 POKES+3,12:POKES+2,129:FORG=0TO5:NEXT:FORT=0TO90:NEXT
12504 POKES+1,36:POKES,85:FORG=0TO80:NEXT:POKES+4,0:RETURN
```

PROGRAMA commodore Magazine COMPROBADO





La revista imprescindible para todo el usuario de
Ordenadores COMMODORE

commodore Magazine

commodore Magazine

AÑO I - Núm. 5 - Julio 1984 - 250 Ptas.

REVISTA INDEPENDIENTE PARA USUARIOS

Programas, juegos y Concurso

Londres: Quinta Feria Commodore

BASIC, version 4.75

OFERTA
SPECIAL DE
TRODUCCION

Aproveche ahora esta irrepetible oportunidad para suscribirse a COMMODORE MAGAZINE. Envíe **HOY MISMO** la tarjeta adjunta, que no necesita sobre ni franqueo. Depositela en el buzón más cercano. Inmediatamente recibirá su primer ejemplar de COMMODORE MAGAZINE y así durante un año (12 ejemplares).



Bravo Murillo, 377.
Tel. 733 96 62
Madrid - 28020

```
INTRODUCE LOS DATOS SIGUIENTES:

● LA ALTURA EN KM? 200
● V0 TANGENCIAL EN M/SEG? 10000
● G DEL PLANETA (TIERRA=9.8)? 9.8
● RADIO PLANETA EN KM (T=6370)? 6370

LA EXCENTRICIDAD (E) ES .652191673

**EL SATELITE ENTRA EN ORBITA ELIPTICA**

SI  E<0  NO HAY TRAYECTORIA,( CAE )
SI  E=0  LA TRAYECTORIA ES CIRCULAR
SI 0<E<1 LA TRAYECTORIA ES ELIPTICA
SI  E=1  LA TRAYECTORIA ES PARABOLICA
SI  E>1  LA TRAYECTORIA ES HIPERBOLICA

QUIERES VER LA TRAYECTORIA DEL SATELITE?
```

```
DATOS PARA LA REPRESENTACION GRAFICA:

QUIERES BORRAR LAS ANTERIORES ORBIT? SI

DENSIDAD DE PUNTOS (DE 1.5 A 5)? 3

ESCALA: 1 (1PIXEL=530KM)
        2 (1PIXEL=265KM)? 1

SENTIDO DE LA TRAYECTORIA (1 Ó 2)? 1

QUIERES ESTRELLAS DE FONDO? SI
```

CONCURSO DE APLICACIONES BAJO EL CALC RESULT

Tras ardua tarea de evaluacion, nuestro jurado opto por premiar esta interesante aplicacion orientada al tema deportivo. Asi pues el premio se va a Gijon, lugar donde reside el autor. De todas maneras en los datos que nos envio olvido adjuntar su nombre al resto de los datos. Esperamos a que los haga llegar a la mayor brevedad posible para acceder al sustancioso premio.

Mi idea se encamina al tema deportivo. Si, aunque parezca mentira o mas bien raro, creo que puede tener una aplicacion interesante.

Todo entrenador de un equipo, con un minimo de nivel competitivo, ha de realizar unas costosas y monotonas tereas de calculo de porcentajes y estadisticas. Cada jugador va generando unos datos mediante sus actuaciones durante la temporada. De esta manera se ve al final si el jugador fue rentable o no para el club.

En mi caso lo he orientado hacia el baloncesto, deporte que conozco un poco y me defiendo en el.

El entrenador de baloncesto ha de conocer todas las cualidades de sus jugadores, para poder adaptarlos a situaciones o partidos determinados.

Igualmente ha de conocer el porcentaje de tiros, tanto de fuera , de cerca como de libres. Tambien de rebotes, ofensivos o defensivos, los balones perdidos o los recuperados, los tapones, etc.

Al final el entrenador puede hacer un juicio global de lo que ha dado de si determinado jugador durante la temporada.

El metodo utilizado para efectuar todo esto es un poco mas dificil de explicar, pero esta dentro de las posibilidades del Calc Result. La ayuda del manual sera relevante para quien no conozcan su manejo.

Antes de nada, hay que adjudicarle una tabla a cada jugador, mediante sus datos particulares: Nombre, Apellidos, Edad, Estatura y Puesto ocupado en el equipo.

Inmediatamente, debajo se incluyen las etiquetas de los datos necesarios a considerar: Tiros, Rebotes, Asistencias...., que se iran llenando despues de cada partido. Una vez terminada la temporada, se efectuarian la suma y el calculo de los porcentajes, siguiendo los metodos previstos en el Calc Result para efectuarlos.

Posteriormente podrian incluirse las estadisticas de cada uno de los datos para poder analizar la regularidad del jugador en cada faceta.

Con los datos obtenidos de cada jugador se puede valorar el conjunto y elaborar una estadistica del conjunto, evaluando el potencial del mismo.

Las posibilidades no quedanb limitadas a los partidos, tambien se puede aplicar el Calc Result a la preparacion fisica de los jugadores, calculando los progresos que van realizando, calculandose el limite final, el del conjunto y el poderio fisco del conjunto.

La cuestion pecuniaria tambien tiene cabida, aunque este no sea un tema puramente deportivo. Se puede llevar la cuenta de lo que gana cada jugador, las primas, etc.

La intencion perseguida por esta aplicacion es ahorrarle cantidad de papeleo al entrenador.

En la figura adjunta se visualiza un ejemplo de esta aplicacion al seguimiento de un jugador durante la temporada.

	Rebote ofensivo	Defensivo	Totales	Tiros fuera	Libres	Cerca	Palmeos
10-X-84	0	1	1	15-20	3-4	5-7	1-2
17-X-84	0	0	0	17-18	2-2	0-0	0-0
24-X-84	0	2	2	10-15	5-7	0-2	0-1
.....	.	.	.	.....	...	...	...
.....	.	.	.	.....	...	...	...
5-V-84	1	3	4	15-16	6-6	3-4	1-1
TOTAL	10	28	38	120-215	81-102	30-60	7-15
%	-	-	-	39	80	30	48

Concurso

Calendario perpetuo

Angel Jiménez Romero, de Córdoba, nos remite este programa desarrollado en un **Commodore 64**. Aunque la idea en que se basa no es excesivamente original, sí lo es en el diseño. Su fin es dar a conocer el día de la semana en que cae una fecha que nosotros determinamos, de cualquier mes y año. También muestra la diferencia, en días entre dos fechas introducidas. Igualmente puede imprimir en la pantalla el calendario de un determinado mes. La única limitación que impone es que el año no sea anterior al cero de nuestra era, aunque si lo quisiéramos hacer, la dificultad desaparece al introducir el año con signo negativo.

Entre otras de sus características hayamos que tiene en cuenta los años bisiestos. Asimismo tiene en cuenta si el año ha de seguir el cómputo Juliano (fechas anteriores al 5-11-1582) o Gregoriano (las posteriores).

Están observadas en el programa protecciones contra fallos en la introducción de datos, por ejemplo si el mes lleva una cifra superior a 12.

```
5 REM.........**..CALENDARIO..**
6 :
10 PRINT"□":DIML(13):B$="     "
15 POKE53280,6:POKE53281,6
20 P=12:GOSUB2000
25 PRINTTAB(15)"■CALENDARIO"
30 PRINTTAB(15)"▬▬▬▬▬▬▬▬▬▬"
40 FORI=1TO7:READD$(I):NEXT
45 DATALUNES,MARTES,MIERCOLES,JUEVES,VIERNES,SABADO,DOMINGO
50 FORI=1TO13:READL(I):NEXT
55 DATA0,31,59,90,120,151,181,212,243,273,304,334,365
70 P=16:GOSUB2000
75 PRINT"■ESTE PROGRAMA OFRECE:":PRINT
80 PRINT"EL DIA  DE LA SEMANA DE CUALQUIER FECHA":PRINT
85 PRINT"EL CALENDARIO  DE CUALQUIER MES ELEGIDO":PRINT
90 PRINT"LOS  DIAS  DE DIFERENCIA  ENTRE LAS DOS ULTIMAS FECHAS INTRODUCIDAS"
92 PRINT:PRINT
95 PRINT"■ELIGE LA OPCION  QUE DESEES Y LUEGO PON":PRINT
96 PRINT"LAS  FECHAS USANDO CIFRAS SEPARADAS POR":PRINT
97 PRINT"COMAS (EJEMPLO: 9,11,1970), Y <RETURN>":PRINT:PRINT:PRINT
98 PRINT,"■PULSA UNA TECLA PARA EMPEZAR■"
100 REM.....DIA SEMANAL Y SI ES BISIESTO
105 DEF FNJ(N)=N-7*INT(N/7)+1
110 DEF FNB(A)=(A/100=INT(A/100))-(A/4=INT(A/4))-(A/400=INT(A/400))
119 :
120 REM..........EMPEZAR
121 :
130 GETX$:IFX$=""THEN130
140 PRINT"□"
147 :
148 REM........ELEGIR OPCION
149 :
150 P=21:GOSUB2000
160 PRINT"■ELIGE: C=CALENDARIO"
180 PRINT"        D=DIA SEMANAL";:POKE204,0
```

```
185 GET X$:IFX$=""THENGOTO185
190 PRINT" ";:POKE204,1:IFX$="C"THENGOSUB300
194 IFX$="D"THENGOSUB200
198 GOTO150
199 :
200 REM.......CALCULO DIA SEMANAL
201 :
205 PRINT"▒▓";
207 FORI=1TO60:PRINTB$;:NEXT:P=1:GOSUB2000
210 INPUT"PON DIA,MES,Y CIFRA ANUAL";D,M,A
215 IFM<0ORM>12THEN200
220 GOSUB2500:GOSUB1000
230 X=FNJ(N)
240 PRINT" EL ";D;"DEL";M;"DE";A;"  ES  ";D$(X)
243 IFN1=0THEN250
245 PRINT"▒ ▒FECHA  ANTERIOR▀"," ▒DIFERENCIA▀",
248 PRINT"   ";D1;"/";M1;"/";A1,"    ";N-N1
250 D1=D:M1=M:A1=A:N1=N:RETURN
299 :
300 REM......CALENDARIO
301 :
302 P=5:GOSUB2000
304 FORI=1TO186:PRINTB$;:NEXT
305 P=9:GOSUB2000
307 INPUT"▓PON EL MES,Y LA CIFRA ANUAL";M,A:D=1
308 GOSUB2500
312 P=5:GOSUB2000
313 PRINTTAB(25)"▓MES:";M;"DE";A
314 IFM<1ORM>12THEN300
315 FORI=1TO7:PRINTTAB(6*I-5)LEFT$(D$(I),3);:NEXT:PRINT
330 B=FNB(A):GOSUB1000
340 X=FNJ(N):GOSUB1500
380 L=6*X-5
390 PRINT"▓";:FORI=1TOT
400 PRINTTAB(L);:IFI<10THENPRINT" ";
410 IFL>34THENPRINT"▒";I;"▀";:GOTO440
420 PRINTI;
440 L=L+6:IFL>37THENL=1:PRINT
450 NEXT:PRINT:RETURN
997 :
998 REM....DIAS TRANSCURRIDOS
999 :
1000 N=365*A+L(M)+D+INT((A-1)/4)+3+(M>2)*(A/4=INT(A/4))
1010 IFA*1000+M*20+D<1582214THENGOTO1050
1020 N=N-11-INT((A-1601)/100)+INT((A-1201)/400)
1050 RETURN
1497 :
1498 REM.....TOTAL DIAS DEL MES
1499 :
1500 B=FNB(A)
1510 T=L(M+1)-L(M)-(M=2)*B:RETURN
1595 :
1995 REM .....POSICIONAR
1999 :
2000 PRINT"▓";:FORI=1TOP:PRINT"▒";:NEXT:PRINT:RETURN
2450 :
2490 REM.......BORRAR LINEA ESCRITA
2495 :
2500 PRINT"TT";:FORI=1TO10:PRINTB$;:NEXT:RETURN
2560 END
```





Cartas

P.: Soy uno de sus lectores permanentes, pues poseo un **Vic-20** y me gusta su revista. Aunque me gustaría que esta carta fuera publicada en la revista, me conformo con que la leáis, pues se trata de un grave problema. En su última publicación, señores, han premiado con la suma de 5.000 pesetas al programa para el **Vic** llamado TEST de un tal Alfredo José Díaz (con el cual yo no tengo nada que ver), pues bien, vamos a la cuestión, el caso es que el programa había ya sido publicado anteriormente en otra publicación nacional con el nombre "Pánico en el fondo del mar".

El programa es exactamente el mismo y yo lo he reconocido, pues lo escribí hace tiempo, y bien esperando que tomen las medidas necesarias contra este "pirata" de programa les saluda atentamente.

L.T.F. Almería

En parecidos términos se expresa otro lector:

P.: ... El único cambio del señor premiado fue ponerle las listas correspondientes a las puntuaciones de los helicópteros, barcos, etc., que esas órdenes las pone mi hijo, que tiene siete años, aclárenme si esta forma de concursar es correcta.

J. Inclán. Gijón

R.: *Pues bien, L.T.F., como nos pides, publicamos tu carta.*

J. Inclán: Efectivamente, ésa no es forma de participar en concurso. Los programas deben ser inéditos y originales, sino cualquier concurso pierde su esencia.

Como es lógico, escapa de nuestro control saber qué programas se publican o no en todas las publicaciones que en el mercado hay. Sin embargo, confiamos en que nuestros lectores nos pondrán al corriente de cualquier irregularidad que observen, como es este el caso.

Cuando ocurre algo de este tipo nos vemos afectados en nuestra buena fe, pues suponemos que los concursantes quieren dar a conocer sus desarrollos. El premio de las 5.000 pesetas no es un fin en sí, sino un acicate y un reconocimiento al esfuerzo de días o tal vez de meses, de ratos libres preparando el programa.

Por lo que al concursante toca, hoy mismo le enviamos una carta en la que pedimos explicaciones sobre el tema. No nos la debe solamente a nosotros, todos los participantes tienen derecho a ella.

P.: Soy un usuario del *Vic-20*, y he descubierto su revista por casualidad. Me he encontrado una sección que me ha atraido mucho: la sección **"CONCURSO"**, pero después me he preguntado, cómo participar en este concurso. Mi pregunta es la siguiente: ¿cómo se han de enviar los programas? ¿Cinta, listado o qué? La revista me parece fabulosa, y espero que guste a los además usuarios de **Commodore.**

Javier Elustondo. Algorta (Vizcaya)

R.: *Muchas gracias por los halagos vertidos a favor de la revista. A juzgar por lo que nos cuenta nuestro departamento de circulación, parece que la revista sí está gustando a los usuarios, por lo menos la compran.*

Aprovechamos esta carta para contestar a otras varias, cuyo contenido es similar. Los programas se pueden enviar a nuestra redacción. No se imponen más trabas que las que afectan a la originalidad y exclusividad. Aclaramos: los programas deben ser inéditos y no haber sido enviados a ninguna otra publicación nacional o extranjera, sean revistas, libros o cualesquiera otras.

Para que los programas puedan concursar es necesario que vengan grabados en cinta de cassette o diskette, indicándonos para qué ordenador han sido desarrollados, *Vic-20, C-64, 8000*, etc. También deberán ser acompañados de una explicación y por supuesto el nombre, dirección y teléfono de contacto (hay que saber a *quien se envia el premio). El listado es aconsejable, pero entendemos que no todo el mundo tiene acceso a una impresora. En caso de dificultad no nos importará listarlo nosotros, de hecho lo hacemos con todos, para mantener la uniformidad de la revista. Decimos que es aconsejable porque durante el transporte, los datos grabados en soporte magnético podrían sufrir alguna alteración total o parcial. Con un listado en la mano costaría menos trabajo reconstruirlo. Por lo demás, el servico de* **Correos** *nos hace llegar los envíos de los concursantes sin mayor problema.*

P.: Soy un "commodoriano" novato. Adquirí mi **C-64** a través de la Caixa... Muchas veces no puedo copiar sus interesantes programas, ya que debido a mi bajo nivel informático, no sé qué teclar debo pulsar en mi **C-64**, para conseguir los signos gráficos que aparecen en los listados.

Me ha ayudado mucho la clave que ofrecen en cada número de su revista, pero en ella no aparecen algunos de los gráficos antes mencionados. Esta es mi petición:

"Podrían en sucesivos números de su estupenda revista ir ampliando la relación: * Cómo se ve. * Cómo se teclea, * Efecto conseguido, para que los novatos como yo podamos disfrutar utilizando "todos" los programas que aparecen en la revista.

M. A. Chamorro
Gerona

R.: *En primer lugar permítenos envidiarte por el precioso lugar donde vives. En efecto, la lista de caracteres gráficos que publicamos no es completa, aunque sí recoge los de más frecuente utilización. De todas maneras hemos elaborado una lista más amplia que publicaremos a partir del próximo número de la revista. No obstante, si no quisieras esperar, casi cualquier libro de los existentes en el mercado sobre ordenadores de* **Commodore** *incorporan este tipo de listas. Así pues podrás elegir alguno de los muchos publicados.*

P.: Les escribo para hacerles una pequeña crítica, y espero que sea constructiva. Hace muy poco tiempo que me he comprado un **Vic-20** y de

casualidad cayó en mis manos su revista. Y la revista en general no está mal, pero en portada dicen ustedes: "revista independiente para usuarios", por ahora todo bien, pero después de hojear su revista de junio, el número 4 me dije: esta revista es para usuarios del **Commodore 64**, y es que está muy claro, hay en este número (4) unos 11 listados de programas, de los cuales 8 son para el **64**, para el **Vic-20** y los otros restantes no se sabe y si ni siquiera necesitan ampliación. Por lo tanto pienso que esta revista no es rentable para el usuario del **Vic**. Ya me dirán ustedes que interés puede tener 250 ptas. por un programa. Espero que tomen las medidas oportunas. Les rogaría me dieran contestación a este carta.

J.C. Fontecoba. Basauri

P.: Solicito, por favor, que añadan más programas para el **Commodore 64**. En su revista hay una gran cantidad de programas para el **Vic-20**, y pocos para el **C-64**.

Se despide atentamente.

J. M. Esparraguera
Figueras, Gerona

R.: *Estas dos cartas son sólo una muestra de algunas más que se reciben en la redacción. Como se puede ver nunca llueve a gusto de todos.*

Procuramos que los números estén lo más equilibrados posible en el reparto de programas. Sin embargo, hay ocasiones en las que tenemos mayor número de programas interesantes para un ordenador que para otro. Entendemos que a cada uno le gustaría ver sólo programas para su modelo, pero ésta es una revista de usuarios. De todas maneras no es tan importante la cantidad de programas como la calidad de los publicados. Realmente sometemos a una dura criba todos los que llegan a la revista y son muchos los que quedan en el camino. Si optásemos por publicarlos todos sólo gozarían quienes disfrutan tecleando sin importarles el cometido del programa.

No obstante creemos que la revista en términos globales tiene interés no solamente por los programas, aunque tampoco creemos que estos desequilibrios sean tan pronunciados.

En una edición anterior se omitió, obviamente por error, el listado de una corta rutina que evita que el símbolo de la interrogación aparezca cuando un programa encuentra una sentencia con INPUT. A continuación no sólo vamos a publicar esa rutina, sino dos más.

En la figura 1 observamos la rutina original, cambiando el contenido de la dirección de memoria 19 (decimal). Después de introducirse el dato, se deja de influenciar sobre ella, poniendo a cero nuevamente su contenido. En el mapa de memoria, esta dirección está definida como un almacenamiento temporal de cadenas, que actúa en forma de pila (apilando bytes).

Figura 1.

```
1000 OPEN 1,0
1010 PRINT"ESCRIBE UN DATO ";
1020 INPUT#1,A$
1030 PRINT
1040 PRINT A$
1050 CLOSE 1,0
```

La rutina de la figura 2 presenta un enfoque diferente. Consiste en abrir un canal de comunicación con el teclado, mediante OPEN. Después se utiliza INPUT#, para recibir información desde el dispositivo numerado con 0 (el teclado). Posteriormente se cierra el canal y continúa ejecutándose el programa con toda normalidad.

Figura 2.

```
1000 POKE 19,1
1010 INPUT"ESCRIBE UN DATO ";A$
1020 POKE 19,0
1030 PRINT
1040 PRINT A$
```

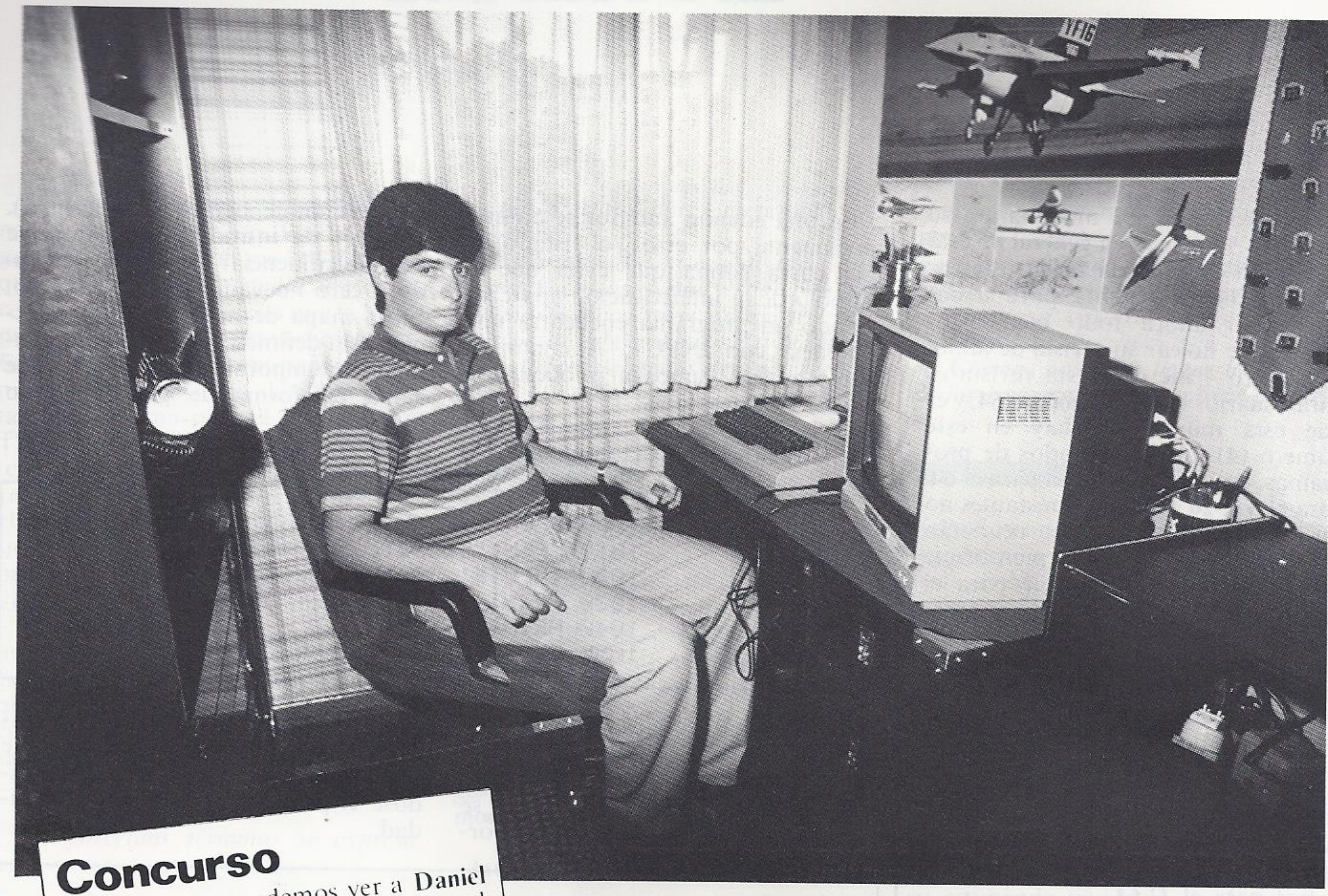
Ambas subrutinas deben introducirse tantas veces como se necesite utilizar INPUT. En el caso presente, las publicamos a modo demostrativo, por eso hemos incluido las líneas con PRINT y PRINT A$, que lógicamente no son necesarias en un programa de aplicación.

La tercera posibilidad difiere de las anteriores. Mediante POKE se alteran los contenidos de diversas posiciones de memoria, que afectan al teclado. La dirección 198 (decimal) contiene el número de caracteres presentes en el teclado. De la 631 a la 634 son parte del *buffer* del teclado. El *buffer* normalmente va de 631 a 640, pero como previamente hemos advertido que en el teclado sólo hay 4 bytes de información, sólo utilizaremos otros tantos bytes del mismo. Si comprobamos en el Manual de **Commodore**, veremos que la información que depositamos hace que el cursor se desplace dos lugares hacia la izquierda y que en su lugar se escriba un espacio (32) y el código ASCII del caracter que deseamos que actúe como *promt* (en este caso se sustituye por un corazón, pero puede utilizarse el espacio o cualquier otro).

Figura 3.

```
1000 POKE 198,4
1010 POKE 631,157
1020 POKE 632,157
1030 POKE 633,32
1040 POKE 634,ASC("♥")
1050 INPUT A$
```

De cualquier modo, esperamos que sean de utilidad.



Concurso

En esta foto podemos ver a **Daniel Cocho** en su lugar de trabajo con el ordenador. Desde él realizó el programa que le hizo acreedor al premio de 5.000 ptas. en el número de junio de **Commodore Magazine.**

Aclaración

Cómo diseñar juegos por ordenador

Por dificultades en la disponibilidad de espacio, hubo que omitir a la hora del cierre el listado del programa de esta serie (figura 7), que debiera haber acompañado al capítulo 4 de la serie.

```
1 GOTO 1000
10 PRINT"▚";
20 FOR J= I TO X
21 PRINT"▮";
22 NEXT
23 FOR J= I TO Y
24 PRINT"▚";
25 NEXT
26 PRINT" ▮";
27 IF D=I THEN PRINT"▮";
28 IF E=I THEN PRINT"▚";
29 IF D=K THEN PRINT"▮";
30 IF E=K THEN PRINT"⌉";
35 PRINT"@";
40 X=X+D
45 Y=Y+E
50 IF X=A THEN D=K
60 IF X=I THEN D=I
65 IF Y=I THEN E=I
70 IF Y=B THEN IF X>(R+K) THEN IF X<R+G THEN E=K
85 IF Y=C THEN GOTO 500
90 IF Y=I THEN IF L(X)=I THEN L(X)=K:P=P+I:PRINT"▮⌉▚▚ ▚▚▚";
```

```
92 IF P=A THEN GOTO 800
95 IF Y=I THEN E=I
100 S=PEEK(Q)
105 S=(NOT S) AND T
110 IF S AND V THEN IF R>G THEN R=R+K+K
120 IF S AND U THEN IF R<F THEN R=R+I+I
130 PRINT"▒";
140 FOR J=I TO C
145 PRINT"▒";
150 NEXT
153 PRINT"■                              ■C]";
155 IF R=Z THEN PRINT"▒▒          ▒▒";:GOTO 10
160 FOR J=I TO R
165 PRINT"▒";
170 NEXT
175 PRINT"▒▒   ▒▒";
180 GOTO 10
500 PRINT"QQQQQ■■■■■ PERDIO "
510 END
800 PRINT"QQQQQ■■■■■ GANO "
1000 REM ** MOVER CARACTERES
1010 POKE 52,48:POKE 56,48:CLR
1020 POKE 56334,PEEK (56334) AND 254
1030 POKE 1,PEEK(1) AND 251
1040 FOR I=0 TO 2047 :POKE I+12288,PEEK(I+53248):NEXT I
1050 POKE 1,PEEK(1) OR 4
1060 POKE 56334,PEEK(56334) OR 1
1070 FOR I=0 TO 7
1080 READ A
1090 POKE 12288+I,A
1100 NEXT I
1110 DATA 60,126,255,255,255,255,126,60
1120 POKE 53272,(PEEK(53272)AND 240)+12
1130 REM ** INICIALIZAR VARIABLES
1140 I=1
1150 A=38
1160 B=20
1170 R=20
1180 K=-1
1190 Y=2
1200 X=20
1210 DIM L(40)
1220 FOR J=1 TO 40
1230 L(J)=I
1240 NEXT
1250 T=31
1260 U=8
1270 V=4
1280 Z=2
1290 F=36
1300 C=21
1310 P=0
1320 Q=56320
1330 D=-1
1340 E=1
1350 G=3
1500 REM ** COLORES DEL FONDO
1510 POKE 53280,0
1520 POKE 53281,0
1530 PRINT "J";:PRINT"▒▒                              ▒▒";
1800 GOTO 10
```

Visualización

Las excelentes capacidades gráficas del **C-64** vienen controladas, todas ellas, por un componente muy importante del *hardware;* el *chip* MOS 6567 *Video Interface Controller* (Controlador del Interface Video), más comúnmente llamado VIC II. Este *chip,* el solito, se encarga de proporcionar todos los modos gráficos de que dispone el **64,** incluyendo entre ellos el modo normal de visualización de caracteres en un formato de pantalla de 25 × 40, el modo gráfico de alta resolución en un formato de 200 × 300 puntos y los *Sprites,* objetos móviles de alta resolución.

La intención de este artículo es entrar a describir en profundidad como se organiza y como funciona uno de estos modos gráficos; el modo normal de visualización de caracteres, que es aquel con el que nos encontramos al encender el ordenador.

ALGO SOBRE EL VIC II Y LA MEMORIA

A pesar de que el **64** dispone de 65536 (ó 64K) posiciones de memoria direccionables, el *chip* VIC II sólo es capaz de acceder simultáneamente a 16K posiciones, es decir, la cuarta parte. Si dividimos los 64K en cuatro bloques de 16K, podremos decirle al VIC II sobre cual de los cuatro bloques (o bancos) va a trabajar, y de esta forma será posible acceder al conjunto de los 64K.

Para saber a cual de los cuatro bancos tiene que acceder, el VIC II mira a una posición de memoria, que corresponde a un registro de un *chip* de Entrada/Salida llamado CIA (6526 *Complex Interface Adapter).* Un par de bits de ese registro, llamados bits de selección del banco, según tengan un valor u otro, van a indicar uno de los cuatro bancos. Estos dos bits son los dos menos significativos (de menor peso) del registro. En la tabla 1 hemos señalado las posiciones de memoria que corresponden a cada uno de los bancos, y también el valor correspondiente de los bits de selección del banco.

Inicialmente, al conectar el ordenador, el VIC II está mirando al banco 0, pero en cualquier momento puede hacerse que cambie a otro banco, para lo que se deben seguir los siguientes pasos:

1. El primer paso consiste en poner a "1" los dos bits de menor peso de la posición 56578 (DD02 en hexadecimal), con lo que ya se puede cambiar de banco.

2. El segundo paso consiste en poner en los bits 0 y 1 del registro de selección de banco el valor correspondiente al banco que queremos

```
10 REM ***************
15 REM * CARACTERES *
20 REM *  GIGANTES   *
25 REM ***************
30 REM
45 FOR I=53248 TO 55297 STEP8: REM *BUCLE PARA LEER ROM *
50 POKE56334,PEEK(56334)AND254: REM *INHIBE INTERRUPCIONES *
55 POKE1,PEEK(1)AND251: REM *DESCONECTA REGISTROS *
60 PRINT"▓"
65 FOR J=0 TO 7: REM *BUCLE PARA LEER 8 BYTES *
70 D(J)=PEEK(I+J)
75 NEXT J
80 POKE1,PEEK(1)OR4: REM *CONECTA REGISTROS *
85 POKE56334,PEEK(56334)OR1: REM *PERMITE INTERRUPCIONES *
90 PRINT"▓▓▓DIRECC.  CONTENIDO CARACTER▓": REM *RUTINA DE IMPRESION *
95 FOR J=0 TO 7
100 PRINT I+J;TAB(10);D(J);TAB(18);
105 FORK=7 TO  0 STEP -1
110 KK%=2↑K
115 PRINT CHR$(032+(134 AND(D(J)ANDKK%)<>0));
120 IFD(J)>KK%THEND(J)=D(J)-KK%
125 NEXTK
130 PRINT
135 NEXTJ
140 PRINT"▓▓▓▓▓▓▓PULSA ESPACIO."
145 FOR T=1 TO 300:NEXTT
150 GETA$
155 IFA$=""THEN150
160 IFA$<>" "THEN STOP
165 NEXT I
```

de caracteres_

seleccionar, según hemos visto en la tabla 1. Este registro está en la posición 56576 (DD00 en hexadecimal).

Para llevar a cabo estas operaciones se utilizan los operadores lógicos AND y OR, en lo que podríamos llamar técnicas de utilización de máscaras y que vamos a ver con un ejemplo.

Si queremos poner a "1" los bits 0 y 1 de la posición 56578 tenemos que hacer un OR con el valor 3. X significa cualquier valor: 0 ó 1.

```
      XXXX XXXX    Contenido de 56578
  OR  0000 0011    Valor 3 en binario
      =========
      XXXX XX11    Resultado
```

De la misma forma, si queremos alterar los bits 0 y 1 de la posición 56576 para cambiar de banco, primero tendremos que ponerlos a "O" con un AND y luego ponerlos al valor deseado con un OR.

```
      XXXX XXXX    Contenido de 56576
 AND  1111 1100    Valor 252 en binario
      =========
      XXXX XX00    Resultado
```

Y ahora para poder acceder al banco 2 haríamos:

```
      XXXX XX00    Contenido de 56576
  OR  0000 0001    Valor 1 en binario
      =========
      XXXX XX01    Resultado
```

Desde BASIC todo este procedimiento que nos permite cambiar el banco de memoria del VIC II se llevaría a cabo mediante el siguiente par de instrucciones.

POKE 56578, PEEK (56578) OR 3, Para poder cambiar de banco.
POKE 56576, (PEEK (56576) AND 252) OR B, Para acceder a un banco nuevo.

TABLA 1

DIRECCIONES DECIMAL	HEXA	BANCO	CIA BITS 1 Y 0	VALOR B
00000 a 16383	0000 a 3FFF	0	11	3
16384 a 32767	4000 a 7FFF	1	10	2
32768 a 49151	8000 a BFFF	2	01	1
49152 a 65535	C000 a FFFF	3	00	0

Donde B es el valor correspondiente al banco, de memoria deseado, según la tabla 1 y que puede ser 0, 1, 2 ó 3.

LA MEMORIA DE PANTALLA

La memoria de pantalla es una zona de RAM (memoria de lectura y escritura) donde se almacena la información de lo que se está visualizando en la pantalla. Como en el modo normal de visualización de caracteres, la pantalla está constituida por 25 filas de 40 caracteres cada una, hay un total de 1.000 caracteres a visualizar. Cada caracter se almacena en una posición de memoria y ocupa un byte, por lo que son necesarios un total de 1.000 bytes de RAM, que constituyen la memoria de pantalla. Estos 1.000 bytes están organizados secuencialmente por filas de forma que el menor de ellos guarda el caracter de la esquina superior izquierda de la pantalla, mientras que el mayor se corresponde con la esquina inferior derecha. Esta correspondencia tan sencilla nos va a permitir escribir un caracter en la pantalla mediante POKE a la dirección correspondiente de RAM y también nos da la posibilidad de leer un caracter de la pantalla mediante un PEEK a la memoria.

Los 1.000 bytes de la memoria de pantalla pueden estar en cualquiera de las 16 zonas de 1K en que se puede dividir el bloque de memoria al que está mirando el VIC II. Inicialmente, al encender el **C-14** la memoria de pantalla se sitúa en el segundo bloque de 1K del banco O, tal y como hemos dibujado en la figura 1, pero en cualquier momento la podemos cambiar, eso sí, diciéndole al VIC II donde queremos ponerla. Esto se hace modificando la posición 53272 que corresponde a un registro del VIC II y cuyos cuatro bits de mayor peso señalan el comienzo de la memoria de pantalla, según la combinación que formen, de entre las 16 que pueden formar. En la tabla 2 están las 16 combinaciones de los 4 bits al lado de la dirección de comienzo que

Visualización

TABLA 2

DIRECCIONES RELATIVAS DECIMAL	HEXA	BITS DE 53272	VALOR B
00000	0000	0000 XXXX	0
01024	0400	0001 XXXX	16
02048	0800	0010 XXXX	32
03072	0C00	0011 XXXX	48
04096	1000	0100 XXXX	64
05120	1400	0101 XXXX	80
06144	1800	0110 XXXX	96
07618	1C00	0111 XXXX	112
08192	2000	1000 XXXX	128
09216	2400	1001 XXXX	144
10240	2800	1010 XXXX	160
11264	2C00	1011 XXXX	176
12288	3000	1100 XXXX	192
13312	3400	1101 XXXX	208
14336	3800	1110 XXXX	224
15360	3C00	1111 XXXX	240

TABLA 3

JUEGO DE CARACTERES	DIRECCIONES DECIMAL	HEXADECIMAL
MAYUSCULAS/GRAFICOS	53248-55295	D000-D7FF
MAYUSCULAS/MINUSCULAS	55296-57343	D800-DFFF

TABLA 4

DIRECCIONES RELATIVAS DECIMAL	HEXA	BITS DE 53272	VALOR B
00000	0000	XXXX 000X	0
02048	0800	XXXX 001X	2
04096	1000	XXXX 010X	4
06144	1800	XXXX 011X	6
08192	2000	XXXX 100X	8
10240	2800	XXXX 101X	10
12288	3000	XXXX 110X	12
14336	3800	XXXX 111X	14

representan. Hay que tener en cuenta que esta dirección es relativa al banco de 16K en que esté trabajando el VIC II: por ejemplo si estamos en el banco 1 de 16K, que comienza en la dirección 16384 (4000 hexadecimal) y situamos la memoria de pantalla en el segundo bloque de 1K, poniendo la combinación 0001 XXXX en la posición de memoria 53272, entonces la memoria de pantalla comenzará en la posición 1024 + 16384 = 17408.

Desde el BASIC es sencillo realizar este cambio de posición de la memoria de pantalla ya que sólo hay que escribir:

```
POKE 53272, (PEEK(53272)AND
15) OR B
```

Siendo B el valor correspondiente al bloque donde se quiere poner la memoria de pantalla según la tabla 2:

CARACTERES EN ROM

Para saber como tiene que dibujar los caracteres en la pantalla, el **64** guarda información, en forma de "ceros" y "unos", referente a cada uno de los caracteres. Esta información se almacena en memoria ROM (memoria no volátil) para que no se pierda cada vez que se desconecta el ordenador, y ocupa un total de 4 Kbytes. Esta memoria se organiza en bloques de 8 bytes, cada uno de los cuales representa un caracter; de esta forma se puede establecer una correspondencia entre puntos de la pantalla y bits, ya que si observamos cómo se dibuja un caracter en la pantalla veremos que está constituido por una serie de puntos individuales y muy pequeños. Estos puntos forman parte de una matriz de 8 × 8 puntos que se asigna a cada caracter. Entonces es muy fácil establecer la correspondencia entre 8 bytes de la ROM y las 8 filas de la matriz, de forma que cada bit "1" del byte representa un punto encendido de la fila oorrespondiente. Así, de esta forma, el **64** sabe como tiene que representar cada caracter en la pantalla.

de caracteres

En la figura 2 puede verse un ejemplo de como está representado un caracter en la memoria ROM y de cómo aparece luego este caracter en la pantalla.

Un aspecto muy interesante de esta ROM es que utiliza toda una zona de memoria correspondiente a registros de Entrada/Salida. La ROM ocupa las posiciones señaladas en la tabla 3, pero ocurre que si miramos el mapa de memoria del **64**, dichas posiciones también corresponden a una serie de registros de E/S, entonces ¿cómo sabe el sistema operativo, o el microprocesador a donde dirigirse cuando aparece alguna de estas direcciones de memoria? Lo que ocurre es que cuando el sistema operativo se encuentra con alguna de estas direcciones, supone que se refieren a los registros de E/S y se dirige a ellos. Sólo cuando se necesita información sobre caracteres, el sistema operativo, con la colaboración del chip VIC II, sabe que tiene que desconectar los registros de E/S y que las direcciones corresponden a la ROM de caracteres. Esta es una idea muy interesante, que permite trabajar con mucha más memoria de la que tenemos, puesto que una serie de direcciones de memoria corresponden a cosas muy diferentes, según intervenga o no el VIC II. Es como si, poniendo un simil, un repartidor tuviera que repartir a dos zonas distintas de una ciudad, pero en las que las calles se llamarán de la misma forma. Entonces, además de decirle a que calle debe ir, habría que decirle a cual de las dos zonas. De esto último es de lo que se encargaría el VIC II.

Existe una forma de acceder directamente a la ROM de caracteres, pero necesita de una serie de pasos para abrirse camino, pues de lo contrario, al intentar leer las posiciones de memoria de la ROM, lo que leeríamos serían registros de E/S. Estos pasos son los siguientes:

1. Hay que inhibir interrupciones. Esto significa que hay que evitar que algún dispositivo de entrada/salida pueda interrumpir a la CPU, pues, como vamos a eliminar toda una serie de registros de E/S, podríamos quedarnos colgados (sin control) si llegara a una interrupción, por ejemplo desde el teclado. Recordemos que una interrupción es una señal que llega a la CPU desde el exterior, y que hace que ésta interrumpa la secuencia normal de intrucciones y salte a ejecutar una rutina específica.

2. En segundo lugar hay que desconectar los registros de E/S y colocar en su lugar la ROM de caracteres, para que al referirnos a las direcciones comunes nos encontremos con la ROM.

3. Ahora ya se puede leer toda la ROM de caracteres directamente le-

```
10 REM **************************************
15 REM * CARACTERES DEFINIDOS POR EL USUARIO *
20 REM *            DEMOSTRACION            *
25 REM *                                    *
30 REM **************************************
35 REM
40 REM
45 S=16:B=3:C=12
50 REM *INICIALIZA POSICION DE PANTALLA
55 POKE53272,(PEEK(53272)AND15)ORS
60 POKE56578,PEEK(56578)OR3
65 REM *INICIALIZA EL BANCO DE MEMORIA
70 POKE56576,(PEEK(56576)AND252)ORB
75 X=C*1024:H=INT(X/256):L=X-256*H
80 POKE51,L:POKE52,H
85 POKE 55,L:POKE56,H
90 PRINT"■■■■■■■    CARGANDO CARACTERES DE ROM A RAM■"
95 PRINT CHR$(142) CHR$(8)
100 POKE56334,PEEK(56334)AND254
105 POKE1,PEEK(1)AND251
110 FORJ=0TO1023
115 POKEJ+12288,PEEK(J+53248)
120 NEXTJ
125 POKE1,PEEK(1)OR4
130 POKE56334,PEEK(56334)OR1
135 PRINTTAB(12);"■ CARAGADOS"
140 REM *INICIALIZA POSICION DEL JUEGO DE CARACTERES
145 POKE53272,(PEEK(53272)AND241)ORC
150 PRINT"■■■■PULSA ESPACIO PARA CARACTERES INVERSOS"
155 GETA$
160 IFA$<>" "THEN155
165 FOR J=X+8 TO X+216
170 POKEJ,255-PEEK(J)
175 NEXTJ
180 PRINT TAB(4);"■■■PULSA ESPACIO PARA LA INVASION"
185 GETA$
190 IFA$<>" "THEN185
195 FOR J=12784 TO12791
200 READD
205 POKEJ,D
210 NEXT J
215 PRINT"■"
220 FOR J=1 TO 280
225 PRINT"■";CHR$(62);"  ";
230 NEXT
235 END
240 DATA36,24,126,219,126,60,60,36
```

yendo las direcciones de memoria de la tabla 3, mediante PEEK, desde lenguaje BASIC. Si no hubiéramos acometido los pasos anteriores, ahora nos encontraríamos leyendo registros de E/S en lugar de la ROM.

4. Al terminar hay que volver a conectar los registros E/S.

5. Y por último, para dejar todo como estaba, hay que volver a permitir las interrupciones.

Acceder directamente a la ROM de caracteres puede ser muy interesante, tanto para utilizar la información que contiene y generar con ella caracteres más grandes o más anchos, o también para definir caracteres nuevos y asociarlos a determinadas teclas, lo que se llama caracteres definidos por el usuario.

Desde BASIC, los pasos que hemos indicado para llegar a la ROM de caracteres se llevarían a cabo con las siguientes instrucciones:

Inhibir interrupciones: POKE 56334, PEEK (56334) AND 254

Desconectar los registros de E/S y conectar la ROM en su lugar:

POKE 1, PEEK (1) AND 251

Volver a conectar los registros de E/S:

POKE 1, PEEK (1) OR 4

Volver a permitir interrupciones:

POKE 56334, PEEK (56334) OR 1

Como dicen que una imagen vale más que mil palabras vamos a ver en la pantalla del televisor, y, con el programa que presentamos como está organizada la ROM de caracteres, y vamos a aplicar para ello todos los pasos que hemos visto, para acceder directamente a ella. El programa que hemos llamado CARACTERES GIGANTES va leyendo de 8 bytes en 8 bytes toda la ROM de caracteres, presentando cada dirección, su contenido y al lado el caracter que se representa con los 8 bytes.

CARACTERES DEFINIDOS POR EL USUARIO

Hemos visto como la información necesaria para la formación de caracteres se almacena en la ROM. Sin embargo es posible trasladar el contenido de esta ROM a la RAM (memoria de lectura y escritura) y hacer que el VIC II acceda a esta RAM para generar los caracteres. Como en la memoria RAM es posible escribir, se pueden modificar los bytes correspondientes a algún caracter de forma que, cuando el VIC II imprima dicho caracter, aparezca en la pantalla algo diferente. Esto es lo que se denomina caracteres definidos por el usuario y permite imprimir todo tipo de figuras, marcianitos, letras góticas, animales, notas musicales, en definitiva cualquier cosa, simplemente diciéndole al VIC II que imprima un caracter, o un conjunto de ellos.

Un juego completo consta de 256 caracteres de 8 bytes, es decir 2 Kbytes de memoria, aunque se puede utilizar un subconjunto de menos caracteres sin ningún problema, si se quiere ahorrar RAM.

Para construir el caracter que se quiere utilizar hay que dibujarlo sobre una matriz de 8 × 8 cuadrados, rellenando los cuadros necesarios.

El resto del proceso, tal y como puede verse en la figura 3 consiste en poner al lado de cada fila el valor en binario que representa, poniendo un "1" por cada cuadrado en negro y un "0" por cada cuadrado en blanco. Por último hay que pasar los valores de binario a decimal, siendo estos valores los que hay que colocar en memoria en lugar del caracter que queremos sustituir.

Ahora nos queda ver donde hay que introducir estos valores, es decir en que posiciones de memoria hay que colocarlos mediante POKE.

El nuevo juego de caracteres puede empezar en cualquiera de 8 posiciones dentro de cualquiera de los bancos de 16K, aunque hay zonas específicas de algunos bancos que no conviene utilizar, ya que las utiliza el sistema operativo. Para decirle al VIC II donde tiene que buscar la información de los caracteres, hay que colocar una combinación adecuada en los bits 1, 2 y 3 de la posición 53272 (D018 en hexadecimal) que corresponde a un registro del chip VIC II. En la tabla 4 aparecen las combinaciones posibles al lado de la dirección de comienzo. Hay que tener en cuenta que esta dirección es relativa al banco en el que esté trabajando, por lo que hay que sumarle la dirección de comienzo del banco como vimos anteriormente.

Desde BASIC, el proceso para decirle al VIC II donde tiene que buscar los caracteres se realiza mediante la instrucción:

POKE 53272, (PEEK (53272) AND 241) OR B

Siendo B como siempre el valor de la tabla 4.

Vamos a presentar a continuación un programa que permite definir caracteres al usuario, y que resume todo lo dicho en este apartado.

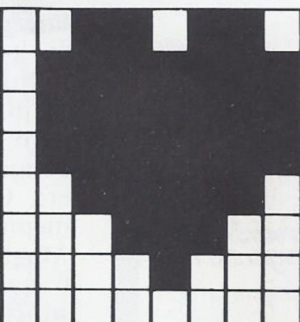
De nuevo hay que inhibir interrupciones, pero antes de ello vamos a engañar al ordenador, poniéndole un límite a la memoria disponible para el BASIC, de forma que un programa en este lenguaje no pueda borrar nuestro juego de caracteres. El juego de caracteres se transfiere de ROM a la zona de RAM que empieza en la dirección 12288, mientras que el tope de memoria para el BASIC lo vamos

de caracteres

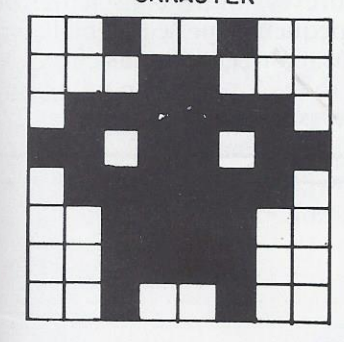
CONTENIDO

DIRECCION	DECIMAL	BINARIO
53912	54	00110110
53913	127	01111111
53914	127	01111111
53915	127	01111111
53916	62	00111110
53917	28	00011100
53918	3	00001000
53919	0	00000000

CARACTER



CARACTER



BINARIO	DECIMAL
0010 0100	36
0001 1000	24
0111 1110	126
1101 1011	219
0111 1110	126
0011 1100	60
0011 1100	60
00100100	36

a colocar en la dirección 12287. Para poner el tope hay que modificar 4 posiciones de memoria que actúan como punteros indicadores y que son:

1. Direcciones 51 y 52, indican la mayor dirección disponible para las variables.

2. Direcciones 55 y 56, indican la mayor dirección disponible para un programa en BASIC.

Para poner como tope en estas direcciones el valor X hay que utilizar las siguientes instrucciones:

```
X = 12287
BA = INT (X/256) Byte alto
BB = X—256+BA Byte bajo
POKE 51, BB
POKE 52, BA
POKE 55, BB
POKE 56, BA
```

El programa incluye muchas de las ideas desarrolladas en el artículo.



Envianos la foto de tu ordenador

En Commodore Magazine hemos pensado que sería buena idea cederos parte del espacio editorial para publicar la foto de vuestro rincón de trabajo. Para ello basta con que nos enviéis cualquier foto en la que se vea, con detalle, como habéis dispuesto vuestra habitación o el comedor de casa. Si preferís aparecer sentados al teclado, tampoco impórta. Es conveniente que acompañéis la foto con unas líneas descriptivas de la instalación y, por supuesto, vuestro nombre.

¡Sorteamos 500 cassettes de juegos para VIC-20 y C-64!

No olvidéis nuestra dirección:

commodore *Magazine*

C/ Bravo Murillo, 377, 5.° A.
Madrid-28020.

Como podréis suponer, nuestra máxima ambición sería conoceros a todos personalmente, saber vuestro interés y ambiciones informáticos. Pero desgraciadamente eso no es posible más que en casos contados.

De momento sabemos que la revista en grandes líneas os gusta. Eso es lo que demostráis mes a mes en los kioskos y con las tarjetas de suscripción.

No obstante es factible tener un conocimiento amplio de vuestras demandas, por un medio frío pero práctico: la antipática encuesta. Queremos conocer vuestras opiniones, qué ordenador tenéis, etc. Con ese mayor conocimiento confiamos en ir mejorando cada día los contenidos de **Commodore Magazine**. Os pedimos que seáis tan amables de tomaros esta pequeña molestia, estamos seguros que su fruto lo notaréis en breve. Somos conscientes de que todo esfuerzo, por pequeño que sea, debe llevar emparejada una compensación. Quinientas cintas de cassette sorteadas entre todos los que respondan nos parece un estímulo interesante. Así que ¡ánimo!

ENCUESTA

1) ¿Tiene ordenador?
a) VIC-20 .... □
b) C-64 .... □
c) Otro .... □

Díganos su marca ....

2) ¿Dónde adquirió su ordenador?
a) Grandes almacenes .... □
b) Tienda especializada .... □
c) Regalo familiar .... □
d) Otros .... □

3) ¿Para qué utiliza su ordenador?
a) Juegos .... □
b) Educación .... □
c) Aplicación .... □

4) ¿Desde cuándo posee su ordenador?
a) Más de 1 año .... □
b) Más de seis meses .... □
c) Más de 1 mes .... □

5) ¿Cuándo lo utiliza?
a) Todos los días .... □
b) Fin de semana .... □
c) Ocasionalmente .... □

6) ¿Con qué frecuencia lee Commodore Magazine?
a) Todos los números .... □
b) Ocasionalmente .... □
c) Soy suscriptor .... □

7) Indique un orden de preferencia entre los temas de la revista:
a) Programas .... □
b) Artículos de divulgación .... □
c) Temas técnicos .... □
d) Cartas al director .... □
d) Otros ....

....

....

....

8) ¿Qué le gustaría encontrar en Commodore Magazine?
a) ....
b) ....
c) ....

9) ¿En qué estrato de edad se encuentra?
a) Menor de 13 .... □
b) Entre 13 y 18 .... □
c) Entre 18 y 25 .... □
d) Entre 25 y 50 .... □
e) Más de 50 .... □

10) ¿Cuál es su ocupación habitual?

....

....

11) Añada aquí sus comentarios

....

....

....

....

CUANDO SE TIENE UN COMMODORE 64 ES MUY DIFICIL SER MODESTO

Cuando se tiene un ordenador personal con 64K de memoria, una magnífica resolución, 16 colores, efectos tridimensionales con "sprites", un sonido equivalente al de un sintetizador, un teclado profesional con 62 caracteres gráficos, toda una amplia gama de periféricos profesionales, la más completa serie de programas educativos, profesionales y de video-juegos...; en resumen, cuando se tiene un ordenador personal como no existe ningún otro en el mercado y el más vendido mundialmente, cuando se tiene el Commodore 64, es muy difícil mostrarlo sin que el orgullo se te note.



EL ORDENADOR PERSONAL DE LA FAMILIA MAS POTENTE

– Sistemas de gestión profesionales series 8000 Y 700. – Ordenador portátil SX 64.
– Ordenador personal COMMODORE 64. – Ordenador familiar VIC 20.



MICROELECTRONICA Y CONTROL, S.A.
c/ Taquígrafo Serra, 7, 5.° BARCELONA-29 c/ Princesa, 47, 3.° G MADRID-8

SLOGAN